Art Department Professional

Version 2.2.0

Serial # 39474447 (on Registration card)

Received June 1993

User Manual

Bundled with A4000, free with purchase of A4000 Mar. 31, 1993.

Software and Manual
by
ASDG, Incorporated
925 Stewart Street
Madison WI 53791-8491
Tel. (608) 273-6585
(see p. 12)

December 14, 1992
Sixth Printing

ART DEPARTMENT PROFESSIONAL USER MANUAL
DECEMBER 14, 1992, SIXTH PRINTING

COPYRIGHTS

Copyright © 1990-1992 ASDG, Incorporated. All Rights Reserved.

The Art Department Professional software and its manual are Copyright © 1990-1992 ASDG, Incorporated. All rights reserved. This document may not, in whole or in any part, be copied, photocopied, reproduced, translated, or reduced to any electronic medium or machine readable format without prior consent, in writing, from ASDG, Incorporated.

The JPEG image compression and decompression technology provided with Art Department Professional is done so under license from Handmade Software, Inc. The Installer program is distributed under license from Commodore-Amiga, Inc. The GT and ASL Emulator libraries are distributed under license from The Dreamers Guild. The DCTV read/write library is distributed with permission by Digital Creations, Inc.

TRADEMARKS

Art Department Professional is a registered trademark and MorphPlus is a trademark of ASDG, Incorporated. Amiga is a registered trademark of Commodore-Amiga, Inc. Caligari Broadcast is a trademark of Octree Software, Inc. DCTV is a trademark of Digital Creations, Inc. Deluxe Paint II Enhanced is a trademark of Electronic Arts, Inc. Digi-View is a trademark of NewTek, Inc. FireCracker 24 is a trademark of Impulse Inc. FrameGrabber is a trademark of Progressive Peripherals, Inc. GIF is a trademark of CompuServe Inc. HAM-E is a trademark of Black Belt Systems, Inc. OpalVision and OpalPaint are trademarks of Opal Technology, Ltd. PCX is a trademark of ZSoft Corporation. Professional Page is a trademark of the Gold Disk, Inc. Microsoft Windows is a trademark of Microsoft Corporation. Rendition is a trademark of Numerical Design, Ltd. Sculpt 4D is a trademark of Byte-By-Byte, Inc. Other marks are trademarks of their respective holders.

DISCLAIMER

ASDG, INCORPORATED ("ASDG") MAKES NO WARRANTIES, EITHER EXPRESSED OR IMPLIED, WITH RESPECT TO THE PROGRAM DESCRIBED HEREIN, ITS QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. THIS PROGRAM IS SOLD "AS IS." THE ENTIRE RISK AS TO ITS QUALITY AND PERFORMANCE IS WITH THE BUYER. SHOULD THE PROGRAM PROVE DEFECTIVE FOLLOWING ITS PURCHASE, THE BUYER (AND NOT ASDG, THEIR DISTRIBUTORS OR THEIR RETAILERS) ASSUMES THE ENTIRE COST OF ALL NECESSARY REPAIR, CORRECTION, OR SERVICING. IN NO EVENT WILL ASDG BE LIABLE FOR DIRECT, INDIRECT, INCIDENTAL OR CONSEQUENTIAL DAMAGE, OR DAMAGES RESULTING FROM LOSS OF USE OR LOSS OF ANTICIPATED PROFITS RESULTING FROM ANY DEFECT IN THE PROGRAM EVEN IF IT HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE. SOME LAWS DO NOT ALLOW THE EXCLUSION OR LIMITATION OF IMPLIED WARRANTIES OR LIABILITIES FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES, SO THE ABOVE LIMITATIONS OR EXCLUSION MAY NOT APPLY.

LICENSE

"ENCLOSED PROGRAM" SHALL BE TAKEN TO MEAN THE SOFTWARE ACTUALLY CONTAINED IN THIS PACKAGE AND ANY SUBSEQUENT VERSIONS OR UPGRADES RECEIVED AS A RESULT OF HAVING PURCHASED THIS PACKAGE.

YOU HAVE THE NON-EXCLUSIVE RIGHT TO USE THE ENCLOSED PROGRAM ONLY ON A SINGLE COMPUTER. YOU MAY PHYSICALLY TRANSFER THE PROGRAM FROM ONE COMPUTER TO ANOTHER PROVIDED THAT THE PROGRAM IS USED ON ONLY ONE COMPUTER AT A TIME. HOWEVER, YOU MAY NOT ELECTRONICALLY TRANSFER THE PROGRAM FROM ONE COMPUTER TO ANOTHER OVER A NETWORK. YOU MAY NOT DISTRIBUTE COPIES OF THE PROGRAM OR THE ACCOMPANYING DOCUMENTATION TO OTHERS EITHER FOR A FEE OR WITHOUT CHARGE. YOU MAY NOT MODIFY OR TRANSLATE THE PROGRAM OR DOCUMENTATION. YOU MAY NOT DISASSEMBLE THE PROGRAM OR ALLOW IT TO BE DISASSEMBLED INTO ITS CONSTITUENT SOURCE CODES. YOUR USE OF THE PROGRAM INDICATES YOUR ACCEPTANCE OF THESE TERMS AND CONDITIONS. IF YOU DO NOT AGREE TO THESE CONDITIONS, RETURN THE PROGRAM, DOCUMENTATION, AND ASSOCIATED PERIPHERALS TO THE VENDOR FROM WHOM THIS SOFTWARE WAS PURCHASED.

THIS LICENSE AGREEMENT SHALL BE GOVERNED BY THE LAWS OF THE UNITED STATES OF AMERICA, STATE OF WISCONSIN AND SHALL INURE TO THE BENEFIT OF ASDG, INCORPORATED OR ITS ASSIGNS.

SPECIFIC RESTRICTIONS

IN ACCORDANCE WITH THE COMPUTER SOFTWARE RENTAL ACT OF 1990, THIS SOFTWARE MAY NOT BE RENTED, LENT OR LEASED.

THIS SOFTWARE AND ITS DOCUMENTATION MAY NOT BE PROVIDED BY A "BACKUP SERVICE" OR ANY OTHER VENDOR WHICH DOES NOT PROVIDE AN ORIGINAL PACKAGE AS COMPOSED BY ASDG, INCORPORATED INCLUDING BUT NOT LIMITED TO ALL ORIGINAL DOCUMENTATION, INSERTIONS, REGISTRATION CARDS, AND SOFTWARE.

PRINTED IN THE USA.

Contents

1 Introduction **1**

1.1 What is Art Department Professional? 1

1.2 Highlights of Version 2 2

1.3 The ADPro Package 3

1.4 Registration and Product Upgrades 3

1.5 System Requirements 4

1.6 Memory Requirements 4

1.6.1 Limiting Memory Usage 6

1.6.2 Reducing Memory Usage 6

1.7 Installation 6

1.8 Starting the Programs 7

1.8.1 ADPro 7

1.8.2 FRED 9

1.9 About This Manual 9

1.9.1 Documentation Conventions 10

1.10 Acknowledgements 11

1.11 About ASDG 11

2 Basic Concepts **13**

2.1 Navigating the User Interfaces 13
2.1.1 User Interface Objects 13
2.1.2 The File Requester 18
2.1.3 The Chooser 20
2.1.4 The Meter Window 22
2.2 Understanding the Image Data Flow 23
2.2.1 Internal Data Types 23
2.2.2 ADPro's Building Blocks 24
2.3 Loading an Image 25
2.3.1 The Load Format 25
2.3.2 The Load Orientation 26
2.3.3 Image Compositing 26
2.3.4 The Image Compositing Control Panel 27
2.3.5 What a Loader Does Internally 29
2.4 Operating on an Image 30
2.4.1 The Operator Type 31
2.4.2 The Visual User Interface 32
2.5 Saving an Image 34
2.5.1 The Save Format 34
2.5.2 Type of Data to be Saved 34
2.5.3 The Save Button 35
2.6 Displaying an Image 35
2.7 Understanding Aspect Ratios 36
2.8 Exchanging Data 37

3 Using ADPro **39**

3.1 Commands 39

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

3.1.1 Load Format 39
3.1.2 Load 40
3.1.3 Save Format 41
3.1.4 Save 41
3.1.5 Orientation 41
3.1.6 Compositing 41
3.1.7 About 42
3.1.8 Exit 42
3.2 Image Operators 42
3.2.1 Operator Format 42
3.2.2 Execute Op 42
3.3 Display Controls 43
3.3.1 ReDisplay 43
3.3.2 Execute 43
3.4 Color Controls 43
3.4.1 Balancing 44
3.4.2 Palette 51
3.5 Screen Controls 63
3.5.1 Horizontal Size 64
3.5.2 Vertical Size 64
3.5.3 Super Hi-Res and VGA 65
3.5.4 Dither 66
3.5.5 Colors 67
3.5.6 A-RES and A-HAM Restrictions 68
3.5.7 Other Overscan Sizes 69

4 Loaders **71**

4.1 ALPHA 71
4.2 ANIM 72
4.3 BACKDROP 73
4.4 BACKLINE 75
4.5 BMP 77
4.6 CLIPBOARD 77
4.7 DPIIE 78
4.8 DV21 78
4.9 FRAMEGRABBER 78
4.10 GIF 80
4.11 HAME 80
4.12 IFF 81
4.13 IMPULSE 81
4.14 IV24 82
4.15 JPEG 83
4.16 MACPAINT 84
4.17 PCX 84
4.18 POINTER 85
4.19 QRT 85
4.20 SCREEN 85
4.21 SCULPT 86
4.22 TEMP 87
4.23 UNIVERSAL 88

5 Savers **91**

5.1 A2410 91
5.2 ANIM 94

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

5.3 BMP 96
5.4 CLIPBOARD 97
5.5 DPIIE 97
5.6 FC24 98
5.7 FRAMEBUFFER 101
5.8 GIF 102
5.9 HAME 102
5.10 HARLEQUIN 104
5.11 IFF 106
5.12 IMPULSE 106
5.13 IV24 107
5.14 JPEG 108
5.15 OPALVISION 110
5.16 PCX 112
5.17 POSTSCRIPT 112
5.18 PREFPRINTER 119
5.18.1 Setting the Preferences Paper Type 125
5.19 QRT 127
5.20 RESOLVER 127
5.21 SCULPT 129
5.22 SEPARATE 129
5.22.1 Using ADPro Separations With Professional Page 132
5.23 TEMP 133

6 Operators **135**

6.1 Apply_Map 135
6.2 Blur 136

6.3 Broadcast_Limit 137
6.4 Collapse 138
6.5 Color_To_Gray 140
6.6 Colorize 142
6.7 Convolve 143
6.8 Crop_Image 144
6.9 Crop_Visual 145
6.10 DCTV 147
6.11 Define_Pxl_Aspect 147
6.12 DeInterlace 149
6.13 Displace_Pixel 149
6.14 Dynamic_Range 150
6.15 Gray_To_Color 151
6.16 Halve 152
6.17 Horizontal_Flip 152
6.18 Intensity_Range 152
6.19 Interlace 153
6.20 KillTemp 154
6.21 Line_Art 154
6.22 Median_Filter 155
6.23 Negative 155
6.24 OpalPaint 156
6.25 Polar_Mosaic 156
6.26 Rectangle 158
6.27 Rectangle_Visual 159
6.28 Rem_Isolated_Pxls 160

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

6.29 Rendered_To_Raw 161
6.30 Roll 162
6.31 Rotate 163
6.32 Saturation 165
6.33 Scale 166
6.33.1 Pixel Aspect 168
6.34 Sim_Print 169
6.35 Spin 170
6.36 Text_Visual 171
6.36.1 Example Usage 176
6.37 Tile 177
6.38 Tile_Visual 181
6.39 TPort_Controller 182
6.40 Twirl 183
6.41 Vertical_Flip 185

7 ARexx Interface **187**
7.1 Addressing ADPro 187
7.2 Getting Results of Commands 188
7.3 Launching ARexx Programs 189
7.4 Understanding the General Rules 190
7.5 Using ARexx Commands 190
7.6 Working With ParseArgs 192
7.7 Using the Supplied ARexx Programs 193

8 FRED **195**
8.1 Sequences, Cells, and Frames 196

8.1.1 Cells vs. Frames 197
8.2 Selection and Arrangement of Cells 197
8.3 Cell Information 198
8.4 Animated Stamps 201
8.5 The ADPro-FRED Link 202
8.5.1 Pre and Post Processing 204

9 AnimOps **207**
9.1 Compositor 207
9.1.1 Projects 208
9.1.2 Components 208
9.1.3 Results 210
9.2 TimeStretch 211
9.2.1 Main Control Panel 212
9.2.2 Extra Options Control Panel 213

10 ADPro Reference **215**
10.1 Starting ADPro 215
10.2 ADPro Control Screen 217
10.2.1 Commands 217
10.2.2 Balancing Controls 221
10.2.3 Palette Controls 224
10.2.4 Screen Controls 230
10.2.5 Other ARexx Commands 233
10.3 Loaders 243
10.3.1 General Information 243
10.3.2 ALPHA 246

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.3.3 ANIM 246
10.3.4 BACKDROP 247
10.3.5 BACKLINE 248
10.3.6 BMP 250
10.3.7 CLIPBOARD 250
10.3.8 DPIIE 251
10.3.9 DV21 251
10.3.10 FRAMEGRABBER 252
10.3.11 GIF 253
10.3.12 HAME 253
10.3.13 IFF 254
10.3.14 IMPULSE 255
10.3.15 IV24 255
10.3.16 JPEG 256
10.3.17 MACPAINT 256
10.3.18 PCX 257
10.3.19 POINTER 257
10.3.20 QRT 258
10.3.21 SCREEN 258
10.3.22 SCULPT 259
10.3.23 TEMP 260
10.3.24 UNIVERSAL 260
10.4 Savers 261
10.4.1 General Information 261
10.4.2 A2410 262
10.4.3 ANIM 264

10.4.4 BMP 265
10.4.5 CLIPBOARD 266
10.4.6 DPIIE 266
10.4.7 FC24 267
10.4.8 FRAMEBUFFER 269
10.4.9 GIF 269
10.4.10 HAME 270
10.4.11 HARLEQUIN 271
10.4.12 IFF 273
10.4.13 IMPULSE 273
10.4.14 IV24 274
10.4.15 JPEG 275
10.4.16 OPALVISION 276
10.4.17 PCX 277
10.4.18 POSTSCRIPT 278
10.4.19 PREFPRINTER 278
10.4.20 QRT 280
10.4.21 RESOLVER 280
10.4.22 SCULPT 282
10.4.23 SEPARATE 283
10.4.24 TEMP 286
10.5 Operators 286
10.5.1 General Information 286
10.5.2 Apply_Map 287
10.5.3 Blur 288
10.5.4 Broadcast_Limit 288

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.5.5 Collapse 289
10.5.6 Color_To_Gray 291
10.5.7 Colorize 291
10.5.8 Convolve 293
10.5.9 Crop_Image 293
10.5.10 Crop_Visual 294
10.5.11 DCTV 295
10.5.12 Define_Pxl_Aspect 295
10.5.13 DeInterlace 297
10.5.14 Displace_Pixel 297
10.5.15 Dynamic_Range 298
10.5.16 Gray_To_Color 299
10.5.17 Halve 299
10.5.18 Horizontal_Flip 300
10.5.19 Intensity_Range 300
10.5.20 Interlace 300
10.5.21 KillTemp 301
10.5.22 Line_Art 301
10.5.23 Median_Filter 302
10.5.24 Negative 302
10.5.25 Polar_Mosaic 303
10.5.26 Rectangle 304
10.5.27 Rectangle_Visual 305
10.5.28 Rem_Isolated_Pxls 305
10.5.29 Rendered_To_Raw 306
10.5.30 Roll 306

10.5.31 Rotate 307
10.5.32 Saturation 308
10.5.33 Scale 308
10.5.34 Sim_Print 309
10.5.35 Spin 310
10.5.36 Text_Visual 312
10.5.37 Tile 315
10.5.38 Tile_Visual 316
10.5.39 TPort_Controller 316
10.5.40 Twirl 317
10.5.41 Vertical_Flip 318
10.6 FRED 318

A Troubleshooting **321**
A.1 Technical Support 321

B Hints and Tips **323**
B.1 ADPro 323

C PREFPRINTER Dither Type Samples **339**

D Additional Utilities **345**
D.1 Splitz & Joinz 345
D.1.1 Amiga Set-Up and Usage 346
D.1.2 Macintosh Set-Up and Usage 348
D.1.3 PC/MS-DOS Set-Up and Usage 349
D.1.4 Windows Set-Up and Usage 350
D.2 ZapDPI 351

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

List of Figures

2.1 An example menu strip hierarchy. 14

2.2 The Operators chooser list. 21

2.3 Two types of meter windows. 22

2.4 The Image Information area, showing the existence of raw and rendered color image data for a 640 x 400 image. . . . 24

2.5 ADPro's Building Blocks. 25

2.6 The Image Compositing control panel. 27

2.7 The Image Operator area, showing the **Operator** and **Execute Op** buttons. 31

2.8 The Crop_Visual control screen, a typical user of the Visual User Interface. 33

2.9 Data flow within ADPro. 35

2.10 Comparison of various aspect ratios. 36

3.1 The main ADPro control screen. 40

3.2 A linear (neutral) color map. 45

3.3 A color map showing an increase in brightness. 46

3.4 A color map showing a decrease in contrast. 48

3.5 A color map showing an example of gamma correction. . . 49

3.6 The Balancing control panel. 50

3.7 The Palette control panel. 52

3.8 The 256 color Palette Editor. 57

3.9 The limited Palette Editor. 61

4.1 The ANIM loader control panel. 73

4.2 The BACKDROP loadercontrol panel. 74

4.3 The BACKLINE loader control panel. 76

4.4 The FRAMEGRABBER loader control panel. 79

4.5 The IV24 loader control panel. 83

4.6 The JPEG loader control panel. 84

5.1 The A2410 saver control panel. 92

5.2 The ANIM Saver control panel. The **Count Frames** button has already been selected (**Number of Frames:** is not zero), causing it to be ghosted. 94

5.3 The first of two control panels for the FireCracker 24 saver. 98

5.4 The second of two control panels for the FireCracker 24 saver. 100

5.5 The ACS HARLEQUIN saver control panel. 105

5.6 The IV24 saver control panel. 108

5.7 The JPEG saver control panel. 109

5.8 The OPALVISION saver control panel. 111

5.9 The primary POSTSCRIPT saver control panel. 113

5.10 Various PostScript metrics. 115

5.11 The second POSTSCRIPT saver control panel. 116

5.12 The third and final POSTSCRIPT saver control panel. . . 117

5.13 The PREFPRINTER control panel (also showing the Additional Options control panel). Note the print direction arrow in the middle of the first page to be printed. 120

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

5.14 Various margins and measures associated with the PREF-PRINTER saver. 126

5.15 The RESOLVER saver control panel. 128

5.16 The SEPARATE saver control panel. 130

6.1 The Blur operator control panel. 136

6.2 The Broadcast_Limit operator control panel. 137

6.3 The Collapse operator control panel. 139

6.4 The Color_To_Gray operator control panel. 141

6.5 The Convolve operator control panel. 143

6.6 The left half of this ray tracing has been sharpened using the Convolve operator. 145

6.7 The Crop_Image operator control panel. 146

6.8 The Crop_Visual operator control screen. 146

6.9 The Define_Pxl_Aspect operator control panel. 148

6.10 The Displace_Pixel operator control panel. 150

6.11 The Dynamic_Range operator control panel. 151

6.12 The Polar_Mosaic operator control panel. 157

6.13 The Rectangle operator control panel. 158

6.14 The Rectangle_Visual control screen and panel showing how a completely filled rectangle is represented. 160

6.15 The Rectangle_Visual control screen and panel showing how a rectangle with a non-zero thickness (but not completely filled) is represented. 161

6.16 The Roll operator control panel. 162

6.17 The Rotate operator control screen. 163

6.18 Components of the Rotate Circle. 164

6.19 The Saturation operator control panel. 166

6.20 The Scale operator control panel. 167

6.21 The Sim_Print operator control panel. 169
6.22 The Spin operator control panel. 170
6.23 The Text_Visual operator control screen and main panel. . 172
6.24 The Tile operator control panel. 178
6.25 A tiling showing no skew. 179
6.26 A tiling showing horizontal skew. 180
6.27 A tiling showing vertical skew. 180
6.28 A tiling showing no skew in which the tiled area evenly fits in the larger bit-map. 181
6.29 The Tile_Visual operator control screen and panel. 182
6.30 The Twirl operator control screen. 184
6.31 Components of the Twirl Circle. 184

8.1 The Set Text Options control panel. 199
8.2 The Change Duration control panel. 200
8.3 The Image Info control panel. 200
8.4 The Invoke ADPro control panel. 203

9.1 The Compositor AnimOp's Components control panel. . . 208
9.2 The Compositor AnimOp's Results control panel. 211
9.3 The TimeStretch AnimOp's main control panel. 212
9.4 The TimeStretch AnimOp's Extra Options control panel. 213

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

List of Tables

2.1 Types of image data that can be merged together. 27

2.2 How mix level affects the new composite image. 28

3.1 The horizontal resolutions supported by ADPro. 64

3.2 The vertical resolutions supported by ADPro. 65

3.3 Color modes and other information. The mixtures of color and display modes marked "A" are possible but require non-standard hardware or an AGA-equipped machine. ADPro allows these modes to be computed but will display them only if the appropriate enhanced hardware is available. 68

4.1 A list of some of the file formats detectable by the UNIVERSAL loader and the file name extensions recognized as hints. 89

5.1 Resolutions supported by the A2410 and the crystals required to implement them. 92

5.2 Dither mask size versus effective color resolution. 121

10.1 Mask values for setting the screen type. 231

10.2 Dither methods and their identifiers. 232

10.3 TIGA screen IDs which may be used as arguments to the B_SIZE parameter. 263

10.4 SCREEN_TYPE mask values for the OPALVISION saver. . . 276

10.5 Values used in constructing a TEXT_STYLE. 314

B.1 Suggested angles and densities for color work run on the Linotronic L300 or L330. 335

B.2 Mix percentages to produce a time lapse effect. 336

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 1

Introduction

This chapter describes the overall Art Department Professional package. After reading it, you will understand:

- what is Art Department Professional
- how to use the manual
- what disks and other materials are included in the package
- why it is important to register your copy of Art Department Professional
- how to receive technical support
- what system hardware and software is required
- how to install the software onto your system

1.1 What is Art Department Professional?

Art Department Professional (ADPro) is an integrated set of powerful image processing tools which facilitate the creation of high quality imagery on the Commodore Amiga personal computer. ADPro's main qualities include:

- It has device and format independent input, output, and processing capabilities. This means that it can read and write nearly any file format, performing any format conversion required. Also, it can control nearly any form of color input or output device, such as color printers, scanners, film recorders, and digitizers.

- It is fully programmable using the ARexx programming language, making it ideal for use in large projects, such as the creation of CDTV applications, animations, presentations, and large publishing works.
- It is exceedingly fast considering the complexity of the processing being done as well as the enormous amounts of data involved. ADPro has been benchmarked against competing products, both commercial and freely redistributable, and was found to be 25 to 250 times faster while performing complicated processing.
- It comes from the company which pioneered color image processing on the Amiga. A company whose only focus is the development of color imaging solutions. And, a company which has been in continuous business in the Amiga market since the computer was introduced in 1985.

1.2 Highlights of Version 2

Following is a brief outline of some of the highlights of version 2.0.0 (and later) of ADPro for the benefit of our continuing customers:

- Several new file formats have been added, including the Amiga's first implementation of the revolutionary JPEG image compression technology. JPEG compression is of enormous benefit to ADPro users who must maintain and work with large collections of images.
- The addition of 24 bit-plane color printing through any Preferences supported color printer means that high quality color proofing is now in the financial reach of all users. This capability also includes the ability to create 24 bit-plane posters larger than outdoor advertising displays!
- ADPro's internal color technology has been completely redesigned for higher precision, greater speed, and smaller memory requirements. Enhanced palette operations now use 24 bits of color precision during all phases of computation, making ADPro's results, already considered among the best in the industry, even better.

  Computing HAM images with 262,144 colors is now possible for use with the HAM-E and other similarly capable enhanced color environments.
- ADPro's scaling technology has been made faster, more flexible, and more precise. It now maintains 64 bits of precision, making ADPro's scaling capabilities the finest we've encountered.
- Several new operators have been added, including one of the first commercial uses of Commodore's scalable font technology added in AmigaOS 2.04, a general purpose convolution operator which gives ADPro users the sharpening capability they wanted, and others.
- Significantly, several operators have been made WYSIWYG to give you finer control and faster manipulations.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

- Several new display boards are now supported for direct viewing of your work including the A2410, Harlequin, OpalVision, DCTV, and others.
- A "universal" loader has been added which allows you greater convenience by automatically detecting the file format of a given file.
- Several user interface changes have been made to incorporate the advantages of AmigaOS 2.04, including true PAL video modes and a faster method of choosing from multiple loaders, savers, and operators.

Again, these are just highlights of the new capabilities in ADPro version 2. With the original ADPro, we earned a dominant place in the color imaging landscape. With the release of version 2 of ADPro, we believe we will earn that distinction again. Your continuing support has made this possible.

1.3 The ADPro Package

The ADPro package comes complete with the following items:

- Three (3) ADPro disks
- This Art Department Professional User Manual
- Registration Card

If you find that you are missing one or more of the above items, please contact the place where you purchased ADPro and ask to exchange your copy with a complete package. If they cannot offer you an exchange, please call us (our phone number is listed in Appendix A) and we will be happy to send you a complete package.

If you find that your version of ADPro does not have a serial number, please contact us by phone and we will send you a properly serialized program.

1.4 Registration and Product Upgrades

ASDG treats the support of its products very seriously. We continually improve our products and, from time to time, make these improvements available in the form of product upgrades.

NOTE It is absolutely imperative that you fill out and return the registration card you received with this product. If you don't, we won't be able to reach you with information about upgrades to ADPro.

Also, technical support will **not** be rendered to anyone who is not in our registered user database. So, please, send in your registration card.

When filling out the registration card, be sure to include the serial number of your copy of ADPro. ADPro's serial number can be found in the first **About** panel, which can be displayed by selecting the **About** button on the ADPro main screen. The registration card must include the serial number to be valid.

When upgrades do occur, you will have to return your original disks along with some amount of money to cover the cost of the upgrade. *International upgrades can only be processed if they are accompanied by a check in U.S. dollars drawn on a U.S. bank or by an international money order.*

Note that if you take advantage of an upgrade offer (i.e., move from one version of ADPro to another), the new version you receive is considered to be exactly the same software as covered by your license agreement. Therefore, you may not sell, lend, lease, or give away your old version as this would be an infringement upon our copyright.

You can sell your copy of ADPro only if: you destroy all copies in your possession, deliver the original disks and manuals to the buyer, and inform ASDG of the transfer of ownership so that the new owner may receive technical assistance from us.

1.5 System Requirements

The Art Department Professional package is compatible with the entire family of Amiga computers. These include the A500, A600, A1000, A1200, A2000, A2500, A3000, and A4000 Amiga computers. All programs (unless otherwise noted) in the ADPro package **require** at least AmigaOS 1.3 to run; they are also compatible with AmigaOS 2.04 and 3.0. For A500, A600, A1000, and A2000 machines, an accelerator is highly recommended.

1.6 Memory Requirements

By their very nature, the Art Department Professional programs require a lot of memory. In fact, ADPro can utilize as much contiguous memory as you have on your machine. While it can run in as little as one megabyte of memory, we suggest a minimum of four megabytes of contiguous expansion memory in order to get any substantial work accomplished. Also, if you use the "high quality" features found in many parts of the program, more memory may be required.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If you use the **SETCPU** program, you should specify the **HEAD** option to allow your 32-bit memory to merge properly with your 16-bit memory. On machines with only 32-bit memory, the **HEAD** option is not required, but specifying it will not cause any problems.

Remember that fast memory must be **contiguous** for ADPro to make full use of it. The more contiguous fast memory you have, the larger the bit-maps (images) you will be able to process.

Ignoring the differences between the file formats that ADPro can process, ADPro really loads only two types of image: color and grayscale.

ADPro defines a grayscale image to be any image in which every color in the image's palette is a shade of gray (i.e., the red, green, and blue contents of each color are equal). ADPro considers anything which is not a grayscale image under this definition to be a color image.

To calculate how much memory would be required for a given color image, you can use the following formula as a guide:

$$MinimumBytesNeeded_{color} = Width_{pixels} * Height_{pixels} * 4$$

This guide will tell you the approximate minimum amount of *contiguous* fast memory (in bytes) required to process a color image of a given size. The multiplication by 4 to give the number of bytes required is actually a short hand for $\frac{32}{8}$. The 32 is the number of bit-planes that ADPro uses for all of its processing. The division by 8 turns the number of bits into bytes (8 bits in a byte).

Take as an example an image measuring 640 by 400 pixels in size. This would require a minimum of 1,024,000 bytes of contiguous available memory to be able to take advantage of all of ADPro's processing capabilities. Of this space, three quarters will be used for 24 bit-plane raw image data. The remaining quarter is used to hold rendered display data.

ADPro converts all color images into 24 bit-plane data when the image is loaded. Grayscale image data is converted into 8 bit-planes rather than 24. This produces a memory requirement rule-of-thumb as follows:

$$MinimumBytesNeeded_{gray} = Width_{pixels} * Height_{pixels} * 2$$

The same 640 by 400 pixel image (as in the previous example) would require a minimum of 512,000 bytes of contiguous memory if it were a grayscale image rather than color.

For more information about ADPro's memory requirements, and for a greater understanding of how ADPro internally stores image data, please refer to Section 2.2.1.

1.6.1 Limiting Memory Usage

On systems with a great deal of memory, you can limit the amount of **fast** memory which ADPro will allocate. By default, ADPro will allocate 128,000 bytes (128K) less than the largest fast memory block available. If you wish ADPro to use less memory than this default, you can specify it in two ways:

1. From the Workbench, you can set a Tool Type (called **MAXMEM**) to the maximum number of bytes that ADPro will be allowed to use for its primary image buffer. For example, specifying **MAXMEM=3000000** will limit ADPro's primary image buffer to 3,000,000 bytes, even if you have 8 megabytes of free contiguous fast memory available.

   Tool Types can be set from the Workbench using the **Information...** command. For more information on how this is done, please consult your Amiga system software documentation.
2. From the Shell, you may specify a command line option of **MAXMEM=***bytes*. For example, specifying the command **ADPro MAXMEM=3000000** will have the same effect as in the Tool Type example.

   The amount of fast memory used will be slightly larger than the amount specified as a memory limit. This extra space is used to hold the program's code in memory, as well as some small data structures.

For more information about Tool Types and Shell options, please consult Section 10.1.

1.6.2 Reducing Memory Usage

ADPro's enhanced palette support consumes as much as 300,000 bytes of memory. If you know that you will not make use of this feature, you can disable it and reclaim the memory it would have used.

From the Shell command line, including the **NOENHANCED** (or the shorter **NE**) option will disable enhanced palette operation.

From the Workbench, you can specify a Tool Type called **NOENHANCED=***n* (where *n* is either zero or non-zero). If set to a non-zero value, enhanced palette operations are disabled. For example, specifying **NOENHANCED=0** permits enhanced palette operations, whereas **NOENHANCED=1** disables them.

1.7 Installation

Insert the ADPro Program Disk 1 into a floppy disk drive. The name of this disk is "ADPro_D1".

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Double-click on the ADPro_D1 disk icon. After the disk's window opens, double-click on the **Install-ADPro** icon. This will start the program that will help you install ADPro onto your floppy or hard disk.

The Installer utility will then execute the installation script. Follow the directions given in the installation to place the ADPro and FRED programs, as well as their support files, where you want them.

If you plan to install ADPro onto floppy disks, select **Expert User** when the Installer asks you to **Set Installation Mode**. This will give you more control over where the files will be installed.

After the Installer has successfully completed the installation, remove the ADPro disk from the floppy drive and store it in a safe place. If your floppy or hard disk somehow becomes unreadable in the future, you will be assured of having an original working copy of the program available to you.

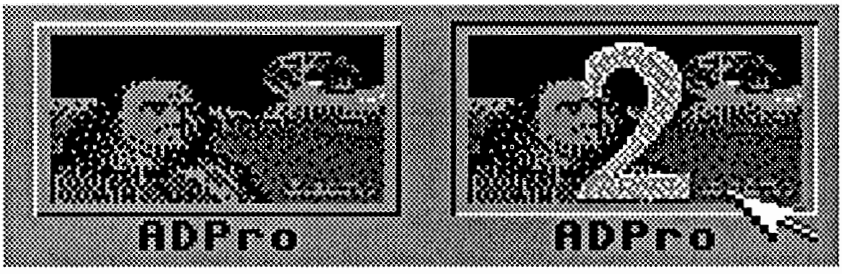
1.8 Starting the Programs

Art Department Professional consists of two programs: the ADPro main program and the FRED program, which allows you to batch process a sequence of images. The ADPro program does not require FRED, but FRED does require ADPro.

1.8.1 ADPro

NOTE All references to Art Department Professional (ADPro) apply to the MorphPlus program, unless otherwise indicated. If you currently own and use ADPro, note that both ADPro and MorphPlus cannot be running at the same time—either one but not both. If you will be using MorphPlus, then disregard this subsection on starting ADPro.

You can start the ADPro program from either the Workbench or Shell.



For Workbench users, open the directory where the ADPro program is located. Locate this icon as shown in the left image above. It should have the name **ADPro** below it. Double-click on this icon to start the ADPro program.

For Shell users, change the current directory to the directory where ADPro is located and start the program with the two commands below. Type each command on its own line, hitting the **Return** key after each line.

```
cd ADPro-directory
ADPro
```

ADPro-directory must be replaced with the actual directory name where the ADPro program and its files were installed. Note that you can prepend the **run** command before **ADPro** above so that you can continue to use the Shell while ADPro is running.

If ADPro is successful in initializing itself, then you will see the ADPro screen. If no screen is displayed, you might have run out of available memory for the program to use or the program could not find a file it needed. In either case, consult Appendix A (Troubleshooting) for more assistance.

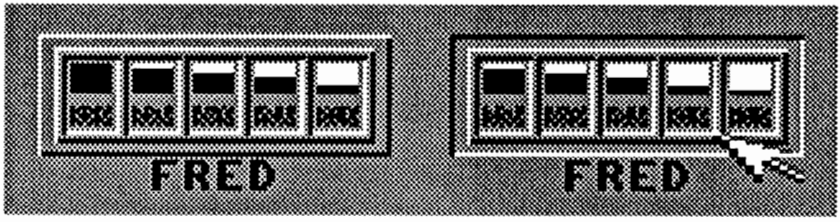
If you start ADPro from the Workbench, disregard this paragraph. If you intend to start ADPro from the Shell, either be in the directory where ADPro is currently located or make sure you have previously executed the command:

```
assign ADPRO: location_of_ADPro
```

before starting the program. This will allow ADPro to find all of its support files. A good place to put this command is in your **s:user-startup** file, if it hasn't already been placed there by the installation utility.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

1.8.2 FRED



For Workbench users, open the directory where the FRED program is located. Locate the left icon as shown above. It should have the name **FRED** below it. Double-click on this icon to start the program.

For Shell users, please start the ADPro program (if it isn't already running) before trying to start FRED. Read the first part of this section for information on how to do this. Note that you should run the ADPro program if you want to start FRED from the same Shell.

From the same shell, type the following command (from the directory where FRED is located):

```
FRED
```

If FRED can initialize itself, then you will see its control screen (which is separate from the ADPro screen). If no screen is displayed, you might have run out of available memory for the program to use. Note that FRED opens a 16 color high resolution interlace screen, which does require more memory to display than the ADPro screen (an 8 color low resolution non-interlace screen).

1.9 About This Manual

The documentation for ADPro is divided into several parts. These are:

Chapter 1 Contains introductory explanations about the manual, the software, and technical support.

Chapter 2 Contains basic concepts you should be familiar with before you can use ADPro.

Chapter 3 Contains the general explanation of how to use the ADPro program.

Chapter 4 Contains the main reference for all ADPro loaders.

Chapter 5 Contains the main reference for all ADPro savers.

Chapter 6 Contains the main reference for all ADPro operators.

Chapter 7 Contains general information about the ARexx interface of the ADPro program. Actual ARexx commands are listed in the master reference chapter.

Chapter 8 Contains the main reference for the FRED program.

Chapter 9 Contains the main reference for all FRED AnimOps.

Chapter 10 Contains a master reference of all of ADPro's control panel commands, keyboard equivalents, menu items, and ARexx arguments.

Appendix A Contains explanations and possible solutions to several problems and questions you might encounter while using ADPro.

Appendix B Contains some helpful hints and tips for getting the most use out of ADPro.

Appendix C Contains sample printouts of the dither types available with the PREFPRINTER saver module.

Appendix D Contains descriptions of the miscellaneous utilities included with the ADPro package.

Index Contains an index for the manual.

The manual is primarily structured as a user manual for you to read from chapter to chapter. However, it can also be used as a reference manual for the program, by using the included master reference in Chapter 10. In the early stages of learning the program, you will use the manual more as a user manual than as a reference. This will allow you to understand how the various parts of the program relate with one another. Once you become more familiar with the program, you will probably use the reference chapter more often for finding quick answers to your questions. ARexx script writers will especially find this reference chapter a necessity.

1.9.1 Documentation Conventions

While reading through this manual, you will see four different types of text styles—plain, **bold**, `typewriter`, and *italics*. These styles are used to denote certain information.

Most of the manual is in the plain style, which denotes regular text with no particular emphasis over other material.

The **bold** style is used to denote labels (text) used in the GUI (Graphical User Interface), such as the **Execute** button on ADPro's main screen.

The `typewriter` style denotes something that you, as the user, would type on the Shell or in one of the input fields on the GUI.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Text written in the **typewriter** style can also mean that it is something you can type from the keyboard. Some examples include **F10**, **Ctrl**, **Shift**, and **Del**.

If a particular equivalent requires two keys be held down, it will be listed as such: **Shift + F1** (with a space between the two keys). Going from left to right, you would hold down each key and press the final (rightmost) key. In the previous example, **Shift + F1** would mean that you should hold down the **Shift** key (either one) and, while still holding down the first key, press the **F1** key.

The *italics* style is reserved, as well, for information that you need to type in. The difference is that information in this style must be defined by you. For instance, in Section 1.8.1 when it asks for *ADPro-directory*, the manual cannot put a directory name in its place because it doesn't know where you actually installed the ADPro program.

The *italics* style is also used to describe variables in mathematical formulae.

1.10 Acknowledgements

ASDG has been assisted by its beta testers and those who have sent in manual corrections and program suggestions. To all of you, we extend our appreciation. We would also like to thank the CDTV community as well as the CDTV staff at Commodore for helping to drive our technology forward by coming up with so many suggestions.

We would also like to thank those of you who have supported our company's efforts by not pirating our software.

1.11 About ASDG

ASDG Incorporated has been a leader in technological innovation and quality since the Amiga premiered. The company has created a number of firsts in the Amiga market, including the first recoverable ram disk, memory boards for the A2000, floppy accelerator, IEEE-488 interface, expansion serial board, and the first 24 bit-plane color image processing system.

In addition, the company has been honored by being the first inductee into Commodore Magazine's "the Amiga Public Domain Hall of Fame," and the company president, Perry Kivolowitz, was given the first Amiga Community Service Award by Amazing Computing magazine.

In recent years we have refined our focus to be the input, processing, and

output of color images. In fact, our ADPro package has become a standard tool among Amiga owners who work with pictures. ASDG was even recognized by Commodore as having made a significant contribution to the development of CDTV (as a consequence of our development of color imaging tools).

We're always looking for quality products which will enhance our product line. Please feel free to contact us concerning our possible publishing or marketing of your color imaging related product.

We can be reached at:

ASDG, Incorporated
925 Stewart Street
Madison, WI 53713
USA

(608) 273 - 6585 (voice)

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 2

Basic Concepts

Before you can take advantage of ADPro's many features, you should learn some basic concepts such as knowing how to use the program's controls, loading, saving, and displaying images, and understanding how your images are represented in memory.

This chapter will take you through the simplest steps involved in understanding these concepts. You will then build upon these concepts in later chapters.

2.1 Navigating the User Interfaces

This section gives general information about various parts of the ADPro's user interfaces. Basic terms such as screens, panels, buttons, and input fields will be described. The use of the file requester, chooser, and meter window will also be discussed.

2.1.1 User Interface Objects

Throughout this manual you will see references to various user interface objects, such as control screens, control panels, buttons, gadgets, input fields, cycle gadgets, rocker switches, and sliders. This subsection will give definitions of each of these, along with their use. The use of keyboard keys will also be discussed.

You should use this as a reference for all user interface terms described in this manual. If a term used in this manual has a different meaning than what will

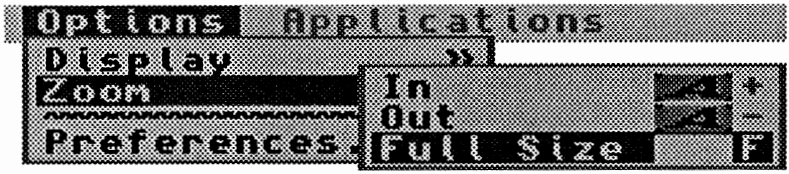


Figure 2.1: An example menu strip hierarchy.

be described below, it will be noted to you.

Control Screens

A control screen (or screen for short) is a large area of the Amiga display. In ADPro, they are used to display the main controls and, for some operators, operator-specific controls. All of the screens have front-to-back gadgets, which allow you to flip between screens. Some have drag bars, which allow you to move your screen up and down to reveal other screens behind it.

Some screens have a menu bar attached to them. You can check if a screen has a menu bar by depressing the right mouse button (menu button). The menu items and subitems can be browsed by holding down the right mouse button, moving the mouse pointer up to the menu bar's (top of the screen) menus, and moving down through the menu items and subitems.

The menu bar has the following hierarchy:

menu
    menu item
        menu subitem

and a sample of this is shown in Figure 2.1.

Menus are the phrases displayed on the menu bar. Menu items are the phrases displayed below each menu. Menu subitems are the ones displayed to the right of a particular menu item (ending with ">>").

To select a menu item or subitem, move the mouse pointer above the item or subitem and release the right mouse button.

Control Panels

A control panel is a smaller area of the Amiga display that can move around on a control screen. Some panels look like standard Amiga windows (as viewable

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

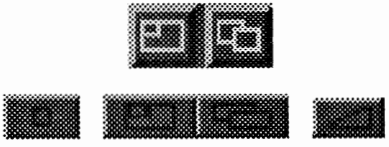
on the Workbench screen), while others have a slightly different 3-dimensional look. Some of these panels can be shrunk to a smaller size and later expanded to full size.

Buttons

Buttons are areas within windows that are outlined by a raised 3-dimensional border. Most of the time a short text string or group of strings is displayed within it. Also, whenever the left mouse button (selection button) is pressed while the mouse pointer is above this area, the button usually changes to a recessed look. In the manual, the terms "press", "select", and "click on" are used interchangeably to describe the selection of a button.

The term "double-click" is used to describe the action of selecting the button twice in rapid succession.

Gadgets



Gadgets are the same as buttons, although they describe the system's buttons—close gadget, zoom gadget, front-to-back gadget, sizing gadget, etc.

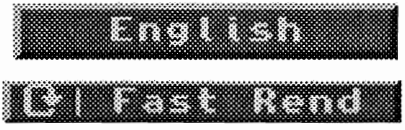
Input Fields

Input fields are areas within windows that either have a recessed 3-dimensional look or a ridged outline around them. They accept input from the user, such as a number of a text string.

NOTE After you finish entering a number or string in one of these input fields, you **must** press the **Return** key for it to be acknowledged by the program.

For some input fields, pressing the **Return** key will move the cursor from field to field. Other input fields require the **Tab** key to be pressed instead. For those types of fields, pressing **Shift + Tab** will move to the previous input field.

Cycle Gadgets and Rocker Switches

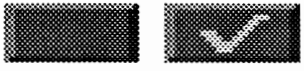


Cycle Gadgets and Rocker Switches are specific, though similar, types of buttons. When selected, they cycle through the possible choices available for that attribute. For example, one of the buttons in the **Screen Controls** area on the main screen lets you cycle between **NonL** (non-interlace) and **ILace** (interlace).

The differences between a cycle gadget and a rocker switch are in the way they look and the way they behave. A cycle gadget uses the Workbench 2.04 or later's imagery—an arrow pointed in a circular, clockwise direction. To cycle forward through the choices, click once on the button. To cycle backward, hold down the **Shift** key while clicking on the button.

A rocker switch looks like a regular button, but it behaves differently depending upon which horizontal half of the button you click. If you click on the right side of the button, it will cycle forward through the available choices. If you click on the left side, it will cycle backwards.

Check Boxes



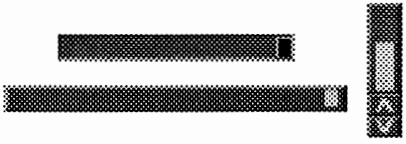
A check box is a button that represents a two-state (on or off, yes or no, etc.) setting. For example, the **MRU** check box in the Chooser (see Section 2.1.3) defines whether the last chosen item from the list will be placed at the top of the list the next time the Chooser is displayed. A check mark will appear on the button for the positive choice (on, yes, etc.), whereas an unchecked box signifies the negative choice (off, no, etc.). Click once on the button to toggle between positive and negative.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Radio Buttons

Radio buttons are a set of two or more oval shaped buttons where each button represents one choice among many choices. These choices are mutually exclusive (only one can be selected at a given time). They are called radio buttons because of their functional resemblance to buttons on a radio, where each button would represent a different radio station. The currently selected choice is displayed as a recessed button. Selecting a different item will recess that item's button and un-recess ("pop up") the previous button.

Sliders and Scrollers

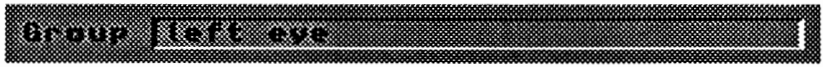


A slider is a gadget that lets you specify a value. A knob (rectangle within the slider area) describes the value in relation to the minimum and maximum values. For example, a slider that can specify a value between 0 and 100 will have its knob in the middle of the slider area for a value of 50. You can also use the left mouse button to select the knob within a slider and drag it to a new position. As you drag the knob, the value of the slider will be updated in real-time in the integer input or text field located to the right of the slider.

A scroller looks similar to a slider, except that the knob describes the position or amount of viewable area within a larger area. For example, the standard Amiga windows have scroller bars (including scroll arrows) to display parts of a disk or drawer's contents. To view other parts of the area, you can drag the slider knob or use the scroll arrows to view the hidden areas in smaller increments.

Sliders and scrollers can either be oriented horizontally or vertically, although most of the time (at least in the ADPro programs) you will see them horizontally.

Text Field / Display Box



A text field, or sometimes called a display box, is an area of a window or panel that contains a word or phrase that conveys some information to you but does not allow you to change it. Often, this field is outlined by a recessed box, signifying that the value (the text) cannot be modified. An example text field is the recessed area immediately below the list of choices in the Chooser list (see Section 2.1.3) that displays the current choice.

Keyboard Equivalents

Some of the operators have keyboard equivalents for frequently used functions. Either one key can be pressed, or a series of keys can be pressed in a particular order. See Section 1.9.1 for a discussion of this.

Also, some control panels have "local" (only usable within the panel itself) keyboard equivalents. They are indicated by an underscore character ('_') underneath the keyboard equivalent. Instead of using the mouse to select, toggle, or activate a particular user interface object, you would simply press the keyboard equivalent.

2.1.2 The File Requester

Anytime you save or load a file, ADPro invokes its file requester. This subsection describes how to use the file requester to specify the name of a file to be loaded or saved.

The ADPro modules use one of two different types of file requesters. The main file requester is described first, followed by the second type.

The REQ File Requester

The REQ file requester, availabled through the **req.library** file in your **LIBS:**, is used for loading and saving files through the ADPro control screen.

A major part of the requester contains the list of files and subdirectories in the currently selected **Drawer**. You can select a file by clicking on its name. The selected filename will appear in the **File** input field (below this area).

Displayed in the right hand list area is the device list. The device list will contain an entry for each of the following types of entities:

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

1. Mounted disk volumes, such as **DFO:**, **RAM:**, or **VDO:**
2. Logical disk names (the name by which each physical disk device is "labelled")
3. Assigned names, such as **L:**, **FONTS:**, and **LIBS:**

Notice that the device list is always kept alphabetically sorted, and that the device list is automatically updated when disks are inserted or removed.

Clicking on an entry in this list changes the **Drawer** specification and displays that drawer's contents in the file list area. If the **Drawer** input field is blank, this indicates that the current directory is the directory containing the ADPro program.

If a directory contains more files and subdirectories than can be displayed in the list area, you can use the scroll bar and arrow buttons (to the right of the list area) to scroll through the current directory's contents.

As the current directory is being read, its contents will be listed in the order they are encountered. This order will almost certainly not be sorted. Should you wish the file list to be sorted alphabetically at any time, simply click the left mouse button on the slider to the right of the file list or upon the scroll arrows above or below the slider. This will instantly sort the contents of the file list. You can also accomplish this from the keyboard by depressing **Alt** + **Return** twice.

To rescan the current directory, you can press the **Get Dir** button.

To exit out of the current directory and back to its parent directory, select the **Parent** button.

Whenever a directory is currently being scanned, you can determine when it is done by watching the status line above the **Parent** button. While it is scanning, you should see a ellipsis (...) at the end of the line. When it is done, the word, **Done**, will replace the ellipsis.

This status line also shown how many files are shown (visible) and hidden. Files can be selectively shown and hidden by changing the text strings in the **Hide** and **Show** input fields. For example, if you enter a file pattern of ***.txt** in the **Show** field, only those files that end in **.txt** will be visible.

You can also show or hide any icon files (ending in a **.info**) by toggling the button above the assigned directory list (at the right side of the file requester).

Another way of selecting the directory (drawer) is to directly type it into the **Drawer** input field. Remember to press the **Return** key so that the file requester can acknowledge your selection.

The **Close** gadget and the **Forget it** button do the same thing—they exit from the file requester without selecting a file. You would do this if you decided not

to select a file at this time.

If you wanted to confirm your choice for a file, select the **OK** button. Alternatively, if the cursor is currently in the **File** input field, pressing the **Return** key will accomplish the same thing.

The ASL File Requester

A major part of the requester contains the list of files and subdirectories in the currently selected **Drawer**. You can select a file by clicking on its name. The selected filename will appear in the **File** input field (below this area).

Also displayed in this list area are all the subdirectories contained within the current directory. Clicking on these entries allows you to examine subdirectories. The subdirectory name will appear in the **Drawer** input field and the files within the subdirectory will appear in the list area.

If a directory contains more files and subdirectories than can be displayed in the list area, you can use the scroll bar and arrow buttons (to the right of the list area) to scroll through the current directory's contents.

To exit out of the current directory and back to its parent directory, select the **Parent** button.

Another way of selecting the directory (drawer) is to directly type it into the **Drawer** input field. Remember to press the **Return** key so that the file requester can acknowledge your selection.

If you want to see all of the available disks and devices, select the **Disks** button. You can then click on one of the listed entries to examine it.

The **Close** gadget and the **Cancel** button do the same thing—they exit from the file requester without selecting a file. You would do this if you decided not to select a file at this time.

If you wanted to confirm your choice for a file, select the **OK** button. Alternatively, if the cursor is currently in the **File** input field, pressing the **Return** key will accomplish the same thing.

2.1.3 The Chooser

Several of the buttons in ADPro allow you to choose one item from potentially many. The current choice is displayed on the button itself. To select a different item, click once on the button to display the chooser.

The chooser opens a window (shown in Figure 2.2) and presents all of the

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

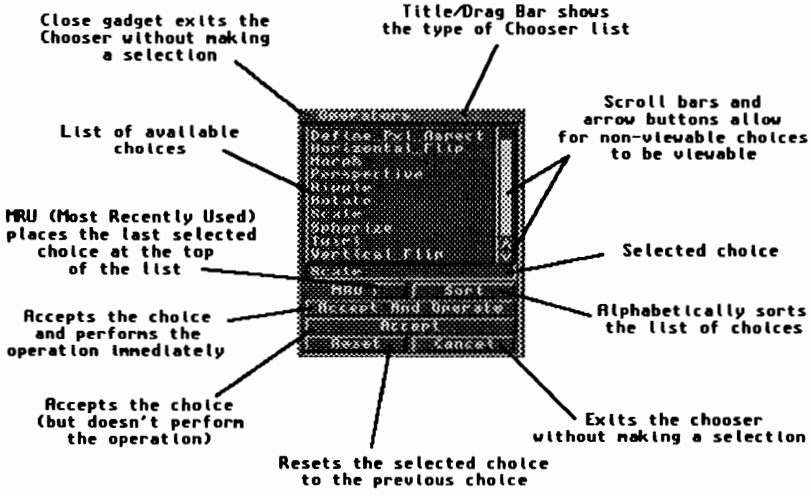


Figure 2.2: The Operators chooser list.

available choices in a scrollable list box. If there are more choices than will fit in the box, you can use the scroll bar or arrows to scroll through additional items in the list. Notice that the bottommost item appears to be in an recessed box rather than an raised box like the rest of the list. This item it not actually part of the scrollable list, but rather displays which item in the list is currently selected.

You can double-click upon any item in the list. If there is a button in the chooser window marked **Accept And** *Something* (for example, **Accept And Save**), then double-clicking on an item in the list will both accept that item and start whatever action is appropriate. If the **Accept And** *Something* button does not appear in the chooser, then double-clicking on an item will simply accept that item but cause no further action.

The items in the list are kept in alphabetical order by default. However, this is not always the case. The checkbox marked **MRU** can exist in two states, either checked or not. **MRU** stands for "Most Recently Used". When checked, the item you selected and accepted the last time you saw this list will be moved from its alphabetical place and placed at the top of the list. This is especially useful when you are toggling back and forth between two or three choices in the list. Using the **MRU** feature, these two or three items will be found right next to each other at the top of the list. This makes using the chooser even faster.

Next to the **MRU** check box is a button marked **Sort**. Selecting this button

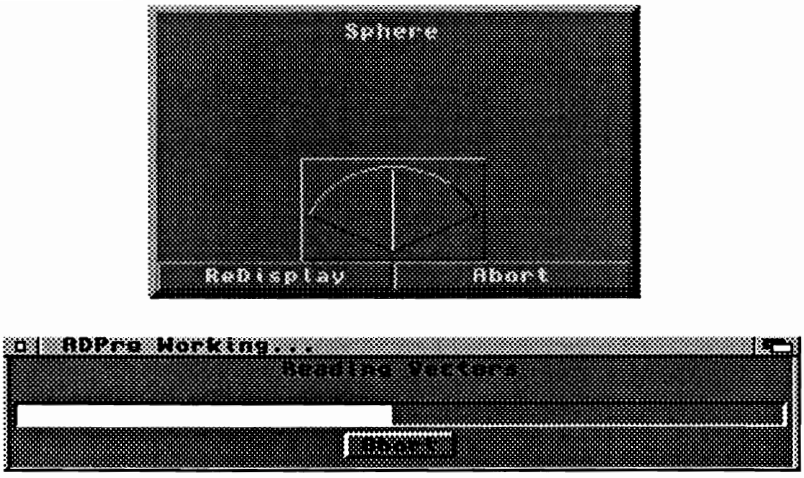


Figure 2.3: Two types of meter windows.

causes the list to become alphabetically sorted if it had gotten out of order due to the **MRU** feature.

There will always be an **Accept** button in the chooser. When there is no **Accept And** *Something* button, the **Accept** button is the same as double-clicking on an item in the list. When there is an **Accept And** *Something* button in the chooser, the **Accept** button causes the selected item to be accepted but no further action taken.

The **Reset** button causes the item which was selected when you last entered the chooser to become active again. The **Cancel** button exits the chooser without making any choice. This is the same as selecting the close gadget at the top left of the chooser window.

2.1.4 The Meter Window

Whenever ADPro is processing your image, one of two styles of meter windows is displayed. Although different in appearance, both display the same information—the amount (percentage) of completeness of a particular operation.

As you can see in Figure 2.3, two general types of meter windows are used in ADPro. The meter at the top mimicks an old-fashioned gas gauge, where the needle displays the progress of the current operation. A description of the

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

operation is displayed above the meter.

The second type of meter window (the bottom image) uses a "level" indicator that extends from the left to right. The description of the operation currently being performed (or of the percentage of completeness) is displayed within the meter itself.

2.2 Understanding the Image Data Flow

Understanding how image data is represented and used by ADPro can be very beneficial for knowing what ADPro is doing with your images. This section will describe how your images are stored, as well as how they "fit" within the image data flow.

2.2.1 Internal Data Types

Within ADPro, image data can take several forms. Understanding the differences between these data types will help you understand how and why ADPro works the way it does.

The first distinction is that image data can be one of two types: *raw* or *rendered.* Rendered image data is data which contains color mapped information. Images such as those which are displayable on the Amiga's screen are examples of rendered image data. For example, an IFF file which contains a 16 color image and can be displayed with any viewing program contains a color map describing the actual color that will be displayed for each value in the image. Without the color map information, the image data can be unrecognizable.

ADPro can create, read, process, or write rendered data with up to 256 color mapped colors. A 256 color rendered image contains 8 bit-planes. Therefore, ADPro can create, read, process, or write rendered data with up to 8 bit-planes.

Raw image data is divided into color and grayscale types. Common to both types of raw image data is that a color map is not required in order to make the image data recognizable. Grayscale raw image data is stored internally in ADPro in 8 bit-planes (24 bit-planes, or 16.7 million shades for color).

Memory permitting, ADPro always converts rendered image data into raw image data during a load operation. The color map of a rendered image file is examined. If it contains only shades of gray, the rendered image data is converted into an 8 bit-plane raw gray image. If any non-gray colors are detected, the image is converted into a 24 bit-plane raw color image.

The **Execute** button causes ADPro to process raw image data into rendered image data.

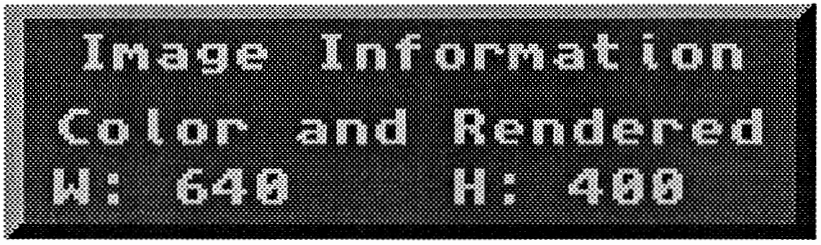


Figure 2.4: The Image Information area, showing the existence of raw and rendered color image data for a 640 x 400 image.

Nearly all of the functions which ADPro can perform require the presence of raw image data. Only one of color or gray raw image data can be processed at one time. Operators are provided to convert between the internal formats of color and gray raw image data.

Some functions in ADPro require only one type of raw image data but not the other. ADPro will display a message if the right type of data is not currently available.

ADPro will display (in the lower left hand corner of the main screen under the heading **Image Information**) the types of image data available. If only rendered image data is available, the word **Rendered** will appear. If only raw color image data is available, the word **Color** will appear. If only grayscale image data is available, the word **Gray** will appear. If both raw and rendered image data is available, both Raw (**Color** or **Gray**) and **Rendered** will be indicated.

ADPro also tells you the width and height of the currently defined image, just below the indication of image data type. This area is called the **Image Information** area. Figure 2.4 shows an the **Image Information** area for a 640 x 400 grayscale image (both raw and rendered image data available).

2.2.2 ADPro's Building Blocks

The beauty of ADPro is that much of the program is modular, made up of separate parts that add additional functionality to the main program. These modules are called Loaders, Savers, and Operators. Figure 2.5 indicates how these modules interact within ADPro.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

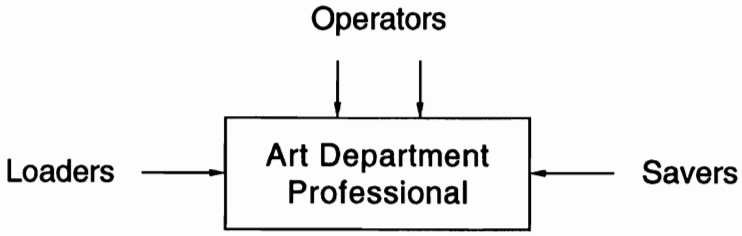


Figure 2.5: ADPro's Building Blocks.

As you can see, a Loader is a source of image data.

A Saver is used as a destination for image data. Savers generally encode raw or rendered image data into a particular image file format for writing to disk.

Finally, Operators are general purpose processing modules which are used to perform the various image processing functions which ADPro supports. Operators are extremely flexible since they allow input, processing, and output of either raw or rendered image data.

2.3 Loading an Image

Before you can work on an image you must first load it into the ADPro program. This section will give a general description of how to do this. If you need more information about a specific loader, please consult Chapter 4.

2.3.1 The Load Format

The Load Format button (shown on the main screen to the right of the **Load** button), displays the type of image data that will be loaded into ADPro the next time the **Load** button is pressed.

To change the type of image data to load, click once on the **Load Format** button. A chooser will appear (if you have enough free memory, otherwise the button will operate as a rocker switch) from which you can select a format. Once you select one, that format's name will be displayed on the **Load Format** button.

2.3.2 The Load Orientation

ADPro can perform a 90 degree counter-clockwise rotation of the image data *during a load operation.* This is accomplished using the **Orientation** button located on the main screen.

Note that the **Orientation** button affects data *only during a load operation.*

Using a combination of the Horizontal_Flip and Vertical_Flip operators and the **Orientation** button, ADPro can produce 90 degree rotations through 0, 90, 180 and 270 degrees.

These can be performed as follows:

0 Degrees	Load the image in the **Port** (portrait) orientation.
180 Degrees	Load the image in the **Port** (portrait) orientation and then perform both a horizontal and vertical flip. The resulting data will be rotated 180 degrees with respect to the data loaded from disk.
270 Degrees	Load the image in the **Land** (landscape) orientation. The resulting image will be rotated 270 degrees with respect to the data residing on disk.
90 Degrees	Load the image in the **Land** (landscape) orientation and then perform both a horizontal and vertical flip. The resulting data will be rotated 90 degrees with respect to the data loaded from disk.

NOTE Although you can use the supplied Rotate operator to rotate to any degree (much more flexible than the previously mentioned orientation tips), these tips are included for when you want to reorient your image when it is being loaded.

2.3.3 Image Compositing

ADPro supports a very powerful image compositing feature which can be enabled by setting the Compositing button to **Comp**. The default value for this button is to disable image compositing. In this case, the button will be marked **Replc** (for Replace).

When the Compositing button is in the **Comp** setting and a load operation is performed, you will be presented with the image compositing control panel. Unless otherwise stated, all loaders support image compositing.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

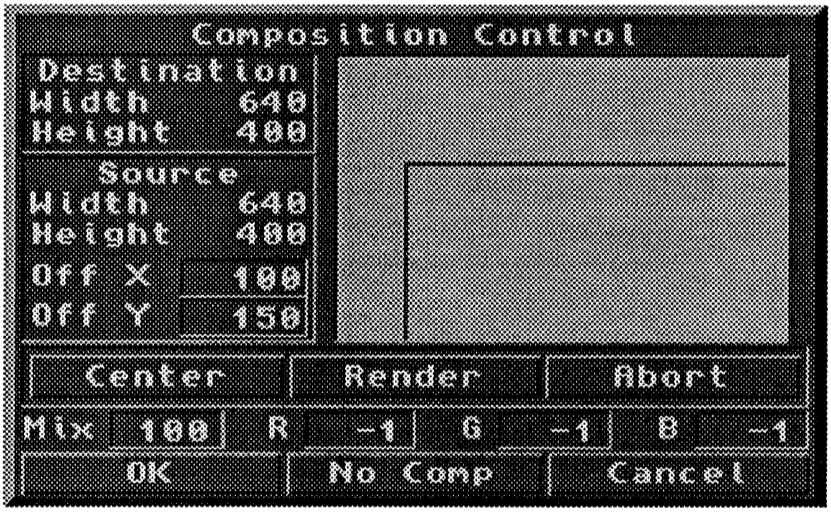


Figure 2.6: The Image Compositing control panel.

Contents Of File	Type Of Raw Data In Memory	
	Gray	Color
Gray	Yes	Yes
Color	No	Yes

Table 2.1: Types of image data that can be merged together.

2.3.4 The Image Compositing Control Panel

If image compositing is enabled, after you select an image to be loaded, the image compositing control panel will appear, as shown in Figure 2.6.

When performing a non-compositing load operation, the loader discards the previous raw and rendered data before loading new data. When compositing, the loader merges the previously defined raw data with the new image being loaded. The exact balance of the merging operation is under your control.

A file containing a grayscale image can be merged (composited) into grayscale or color raw image data. A color image may *not* be directly merged into gray scale raw image data. Table 2.1 summarizes this restriction.

To merge a color image into a grayscale image, first convert the logical structure of the grayscale image into that of a color image using the Gray_To_Color

Mix Level	Weight Of Pixels From Loaded Image	Weight Of Pixels From Previous Image
100	100	0
50	50	50
25	25	75

Table 2.2: How mix level affects the new composite image.

operator.

In the Compositing control panel shown in Figure 2.6, the **Destination** sizes correspond to the width and height of the raw image currently in ADPro, into which the file to be loaded will be merged.

The **Source** width and height denotes the full size of the image to be loaded. The offsets (x and y, labelled as **Off X** and **Off Y**) give the position of the top left corner of the image to be loaded relative to the top left corner of the destination image. An offset of `0,0` means that the top left corner of both the source and the destination will coincide.

Typing offsets into the supplied input fields can be used to set the offset of the top left-hand corner of the image to be merged relative to the top left-hand corner of the original image. Note that negative offsets are perfectly acceptable.

The mix level, which can range from 0 to 100, can be set by typing the desired value to be used in the supplied input field (next to the label that says **Mix**). It represents the weighting the pixels in the image to be loaded should have when being averaged with pixels from the previous (currently in ADPro) image. Table 2.2 shows how various mix levels affect the weighted average between these two images.

So, for example, a mix level of 100 means that the new image data is given a weighting of 100 percent. Thus, the new pixel completely replaces the old pixel where the new image and the old image overlap. A mix level of 50 means that you will get a straight arithmetic average between new and old pixels.

The labelled **R**, **G**, and **B** input fields are used to specify a 24-bit color value which will be regarded as completely transparent during the image compositing operation. One hundred percent of the old image data will be preserved for each pixel in the newly loaded data which matches the color specified in these gadgets.

When compositing a grayscale image, the value in the **R** input field is used for the transparency operation. A value of -1 in any of the input fields disables

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

the transparency check.

Selecting the button marked **Center** causes the image now being loaded to be centered within the backdrop (previous) image. If the image now being loaded is larger than the backdrop image, its center will coincide with the center of the backdrop image.

Selecting the button marked **Render** causes the background image (the image already in memory) to be drawn in grayscale. This will assist you in determining the composition offsets. In order to display the foreground image (the one you are about to merge) in the same way, ADPro would have to read that image twice (once to create the mock-up for positioning, and once to actually load the image data). Loading an image from disk twice was thought to be too expensive (in terms of your time), therefore, this capability is not provided. While the background mock-up is rendering, you can stop it using the button marked **Abort**.

The button marked **OK** will merge the specified image into the currently defined raw image. The button marked **Cancel** will abort the compositing as well as the load operation.

The button marked **No Comp** will continue with the load operation but not perform compositing. This will cause the raw image to be completely replaced by the newly loaded data as if you had disabled image compositing.

Typing **Shift + Return** (pressing the **Return** key while holding down either **Shift** key) is equivalent to selecting the **OK** button. Pressing the **Return** key while the cursor is in either of the offset input fields will cause the cursor to alternate to the other input field. Pressing **Return** while the cursor is in any of the transparency or mix input fields will cause the cursor to appear in the next input field in the sequence. Typing **Alt + Return** causes the cursor to alternate between the offset fields and the mix and transparency fields.

With creative use of the image compositing capability, a myriad of special effects can be performed. For example, you can use a combination of a standard Amiga paint package and the image compositing function in ADPro to perform true color (24 bit-plane) image masking. Specifically, you can do things such as remove just one person from a scan of a group photo and place that person into the middle of a 24 bit-plane ray tracing.

2.3.5 What a Loader Does Internally

This section describes one of the most significant steps that loaders perform when reading in an image. If the file being loaded contains rendered image data (as opposed to true color or raw image data), that data will be converted into 24 bit-plane color raw image data inside ADPro (if the image being read is

rendered grayscale, it will be converted into 8 bit raw grayscale inside ADPro).

Note that even a tiny 1 bit-plane image containing, for example, only red and green, will be converted into 24 bit-planes of raw data. That is, the space required to process an image can be many times larger than the space needed by the image while it is on disk.

If there is not enough memory to convert a rendered image back into raw image data, or if you abort the conversion, no raw data will be available for you to manipulate within ADPro. Many of ADPro's advanced image processing functions require the presence of raw image data in order to operate.

However, there are some important functions you can still perform even when there is no raw image data available (either because you didn't allow it to be created or because it wouldn't fit into available memory).

If the rendered image data you just loaded is displayable by the Amiga, you can view the image directly by hitting the **ReDisplay** button. Also, you can immediately save the image out into a format which supports the same type of rendered data (i.e., IFF-ILBM or IFF-ANIM format).

From ARexx, you can force rendered image data to *not* be converted into raw image data by using the `NOPAD` loader option. This capability will be mentioned in the description of the Load Format ARexx interface.

2.4 Operating on an Image

Once the image has been loaded into ADPro, various manipulations can be performed on it. These manipulations come in the form of operators. The ADPro package comes with several operators, all of which are described in Chapter 6.

As can be seen in Figure 2.5, operators have complete freedom to read, process, and rewrite raw image data maintained within ADPro.

NOTE It is important to note that unlike changes to the raw or rendered data made by any of the screen or color controls, changes made by operators to raw image data cannot be undone without reloading the raw image data.

Also, operators are not, in general, fully interruptible. Since operators actually modify the raw data maintained by ADPro, interrupting an operator in mid-operation will leave partially operated-upon data in memory or may, in some cases, cause all data to be lost (if interrupted).

Operators are separate programs which are run by ADPro when you select the

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Figure 2.7: The Image Operator area, showing the **Operator** and **Execute Op** buttons.

Execute Op button located in the **Image Operator** area on the main screen. These programs must reside in a specific directory so that ADPro can locate them at run-time. The Installer utility that comes with ADPro automatically creates a directory called `Operators2` in the same directory in which ADPro is installed.

If you execute ADPro from the Workbench screen, ADPro can automatically locate the `Operators2` (as well as the `Loaders2` and `Savers2`) directory because it is in the same directory as ADPro.

However, if you want to execute ADPro from the Shell, you must have previously executed the command:

assign ADPRO: *location_of_ADPro*

This will allow ADPro to properly find the `Operators2` directory.

2.4.1 The Operator Type

In the lower left corner of the ADPro screen, you'll see an area labelled **Image Operators** (shown in Figure 2.7). The **Operator** button (above the **Execute Op** button) is used to select which operator will be invoked when the **Execute Op** button is depressed.

Clicking on the **Operator** button will display a list of available operators. You would select the operator to use from this list. See Section 2.1.3 (The Chooser) for instructions on how to select an item from this list.

Once you have selected the operator, selecting the **Execute Op** button will cause the selected operator to execute. Alternatively, you can double-click on the operator entry in the chooser list to automatically execute it.

2.4.2 The Visual User Interface

Almost all of ADPro's operators use a WYSIWYG interface. WYSIWYG stands for "what you see is what you get". WYSIWYG user interfaces are characterized by a very strong visual component (hence the name "visual user interface") with immediate feedback given to the user. The different modules within ADPro which use the visual interface have common features and operating characteristics. This subsection describes these attributes.

Figure 2.8 shows the control screen for the Crop_Visual operator, which is typical of the operators that use the visual interface.

First you will note that operators which employ the VUI open their own screen in front of the ADPro screen. Contrast this to the non-VUI operators, such as Scale, which open a small panel on top of the main screen.

The VUI wouldn't be very visual if an image didn't play a prominent role in the screen. In Figure 2.8, you'll note that the entire left hand portion of the screen is used to display an image of the area being worked upon. This image is displayed in 8 shades of gray and is dithered to enhance the quality of the image. We chose to convert the image "on the fly" to grayscale, since we have such a small number of colors to work with on a standard Amiga display. Being usable on the lowest common Amiga configuration was a primary design goal in developing the VUI.

Note that as the image is being drawn, you can generally interrupt the rendering either by clicking on the button (or selecting the menu item, if one is available) marked **Abort** or hitting **Shift + a** (pressing the **a** key while holding down either **Shift** key).

ADPro will always scale the image so that either the height or the width will fill the screen. This allows you to see as much of the image as is possible, as large as possible. Since the Amiga's display is wider than it is tall, images with a horizontal aspect will "fill" the screen better than images with a vertical aspect. In Figure 2.8, you'll note that the image fills the height of the screen but does not fill the width. This is because the portion of the image on-screen has an overall vertical aspect.

Note that the VUI takes into account the current settings for X and Y resolution. For example, if the X resolution were set to 100 DPI and the Y resolution were set to 300 DPI, a bit-map 100 by 100 pixels in size would be drawn scaled as a stretched out rectangle. While the IFF format supplies an X and Y res-

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

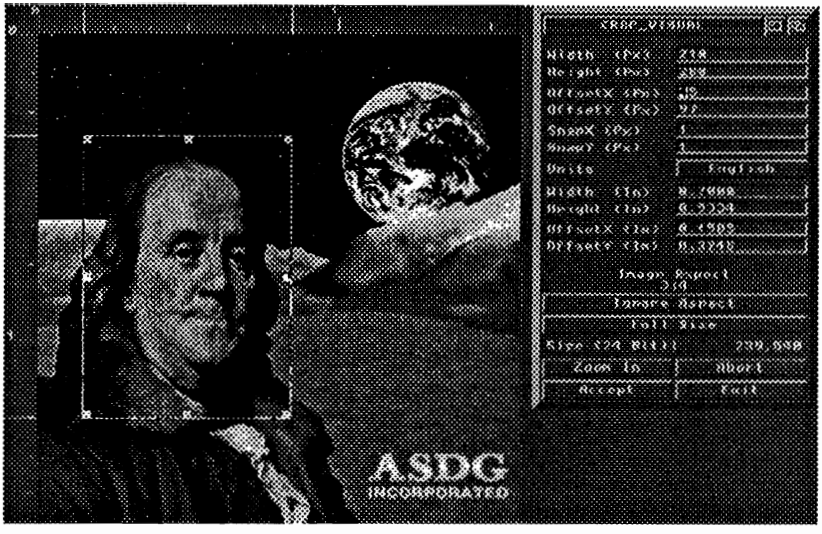


Figure 2.8: The Crop_Visual control screen, a typical user of the Visual User Interface.

olution to ADPro as the image is loaded, you may need to set these values yourself occasionally. See Section 6.11 for more information about setting the X and Y resolution for a given picture.

The ADPro screen, as well as the Visual (Collapse, Crop_Visual, Polar_Mosaic, Rectangle_Visual, Rotate, Spin, Text_Visual, Tile_Visual, and Twirl) operator control screens, will initially try to open up on 8 bit-plane grayscale screens if running on an AGA machine. If it cannot open on this deep bit-plane screen, it will revert to the current 3 bitplane (8 shades) screen.

What this means is that the Image Composition control panel and the above listed operator's preview images will display in 256 shades of gray if at all possible. This feature will take advantage of the added capabilities of AGA-equipped machines, giving you a better representation of your image.

Note that an 8 bit-plane screen will not be as responsive as a 3 bit-plane screen.

Consult the description of the tool types for information on how to suppress the 8 bit-plane display.

Along the top and left borders of the screen you will find a set of rulers. These rulers demark inches or centimeters, depending upon which unit of measure you have selected. These measures, though, are dependent upon ADPro's knowledge of the resolution of the image. The rulers do not break down into fine detail. In inches, for example, tick marks are drawn only down to the

$\frac{1}{8}$ inch level. More accurate measuring information is given in the operator's control panel.

To complete our tour of the typical VUI control screen, let's move to the top of the control panel. The button-like area at the top, in this case labeled **CROP_VISUAL**, is the drag bar for the window. By depressing the left mouse button in this area and holding it down, you can drag the entire window around the screen. You can't drag it off the screen, however. Some operators may use a different style of panel. Specifically, the panel looks like a normal window like those found on your Workbench screen. The drag bar for these windows is also at the top of the panel.

Most of the VUI operators' control panels have a "zip" gadget. You can zip the panel to a smaller size so that you can see more of the image. When the control panel is small (or zipped), you can still manipulate the objects on the image with your mouse. Some operators implement keyboard shortcuts for various commands which can be used while the control window is zipped; they are described in the specific sections in Chapter 6.

Finally, there is the front/back gadget for the screen. Use this as you would any other screen front/back gadget.

2.5 Saving an Image

When you have an image that you like to keep, you should save it to disk. You have the choice of saving it out in one of a few formats. This section will give a general description of how to do this. If you need more information about a specific saver, please consult Chapter 5.

2.5.1 The Save Format

The **Save Format** button, shown on the main screen to the right of the **Save** button, describes the types of image data that can be saved from ADPro and that will be save the next time the **Save** button is pressed.

As with **Load Format**, to change the type of image data to save, click once on the **Save Format** button and select the format from the chooser list. Once you select a saver, its name will be displayed on the **Save Format** button.

2.5.2 Type of Data to be Saved

Recall from Section 2.2.1 that ADPro can maintain up to two different types of image data internally, Rendered and Raw. Most savers will give you an

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

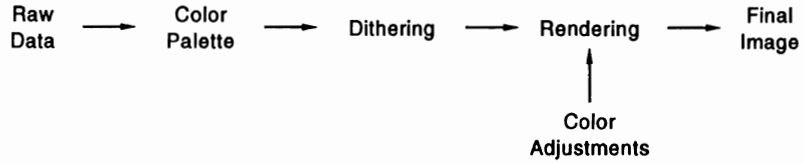


Figure 2.9: Data flow within ADPro.

opportunity to select which type of image data you wish to save. Each saver knows what type of image data it will accept. If you do not have the appropriate type of data available at the time the saver is executed, it will alert you to this requirement.

2.5.3 The Save Button

The **Save** button actually initiates the save operation. It will try to save out the current raw or rendered image data in the format you have specified in the **Save Format** button.

2.6 Displaying an Image

To display your 8 or 24-bit image data, you must create rendered image data in an Amiga displayable screen format. Due to the standard Amiga's graphics display limitation, you cannot view these 8 or 24-bit images directly on your Amiga monitor. For most current Amigas, the most colors you can display is 4096 (HAM mode). Amigas installed with the Advanced Graphics Architecture (AGA) chip set will be able to display more colors. Consult Table 3.3 for a complete listing of available colors in various screen modes.

To create rendered image data, press the **Execute** button (located at the lower right of the main control screen). The raw image data will be analyzed and the image will start to render (display), assuming you have selected your screen controls properly. Figure 2.9 shows how ADPro will process the raw image data into rendered image data.

If your image is wider or taller than the viewable screen area, your image will appear to overwrite itself. This is not actually happening. It is only used to indicate that the image is larger (either wider or taller, or both) than the viewable screen area. Once the rendering has stopped, you can view hidden areas of your image by using the cursor keys. See Section 3.3.1 for more

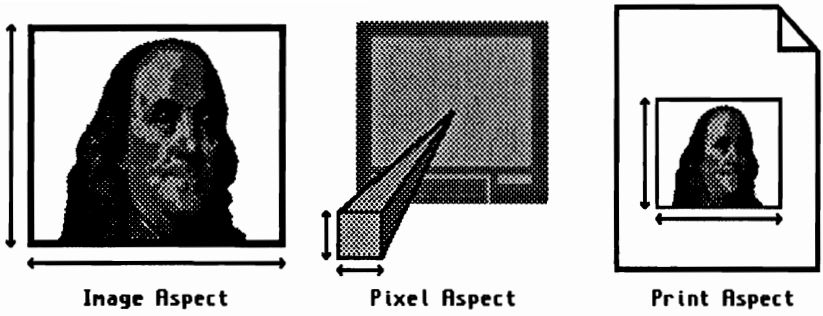


Figure 2.10: Comparison of various aspect ratios.

information on viewing the rendered image.

To return to the main screen, click once on the left mouse button.

2.7 Understanding Aspect Ratios

This section will explain and clarify two types of aspect ratios which you might have heard about before—image aspect and pixel aspect.

An aspect ratio is the relationship between the width of an item to its height, expressed in the form:

$$width : height$$

For example, a 1 : 1 aspect ratio means that the item (whatever is being measured) is as wide as it is high.

Aspect ratios are described by saying their width value to their height value. Taking the previous example, you would say that it has a "one-to-one" aspect ratio.

Image aspect refers to the aspect ratio of the image itself. The *width* and *height* correspond to the width and height (both measured in pixels) of the image, as shown in the leftmost image in Figure 2.10. For example, a 640 x 400 image has an image aspect ratio of 640 : 400 (or more simply, 8 : 5).

Pixel aspect refers to the aspect ratio of an individual pixel on your display device (i.e., your Amiga monitor). The *width* and *height* correspond to the width and height of a single pixel, as visualized in the middle image in Figure 2.10.

 Copyright © 1992 ASDG Incorporated Unauthorized Duplication Prohibited

This ratio is different for each display device, and can also be different for similar display devices if their horizontal and vertical adjustments are different.

2.8 Exchanging Data

Exchanging image data with other computers is divided into two steps:

1. The ability to read and write image files in the logical format used on other computers.

   The ADPro (and Professional Conversion Pack) loader and saver modules solve this requirement by allowing you to read and write files logically structured in the BMP, PCX, and JPEG formats, to name a few. The companion (but separate) product called the Professional Conversion Pack (PCP) includes the TIFF (miscellaneous platforms, specifically Apple and IBM), TARGA (principally IBM), and the Rendition (miscellaneous platforms) formats.
2. Physical connectivity, or how to get the information physically from one computer type to another.

This problem can be solved in a number of ways and is not addressed by this package. One low-cost solution that works quite well for exchanging data with IBM PC type computers is a product called CrossDOS from Consultron ((313) 459 - 7271). It will allow IBM 720K 3.5 inch disks to be used on the Amiga as if they had been native Amiga disks. Also, Workbench 2.1 and later include the ability to read from and write to these same 720K disks. Used in conjunction with these products, ADPro can directly read and write IBM 720K diskettes.

Other solutions to the connectivity issue, based upon network connections, are slow in emerging but making steady progress. At the very least, a serial port based connection made via modem or null-modem will do the job.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 3

Using ADPro

This chapter will describe all aspects of using ADPro from its graphical user interfaces, or GUIs. Detailed descriptions are included for the ADPro's Commands, Loaders, Savers, and Operators.

As you can see in Figure 3.1, the ADPro control screen is made up of six areas: Commands (upper left set of buttons), Color Controls, Screen Controls, Image Operators, Image Information, and Display Controls. The first three areas and the Display Controls area will be described in this chapter; Image Operators and Image Information were discussed in Section 2.4.1.

3.1 Commands

This section describes all of the commands (shown in the **Commands** area) available from the ADPro main screen.

3.1.1 Load Format

The **Load Format** button displays the current image data format that will be loaded the next time the **Load** button is pressed. For a listing of available formats, please see Chapter 4.

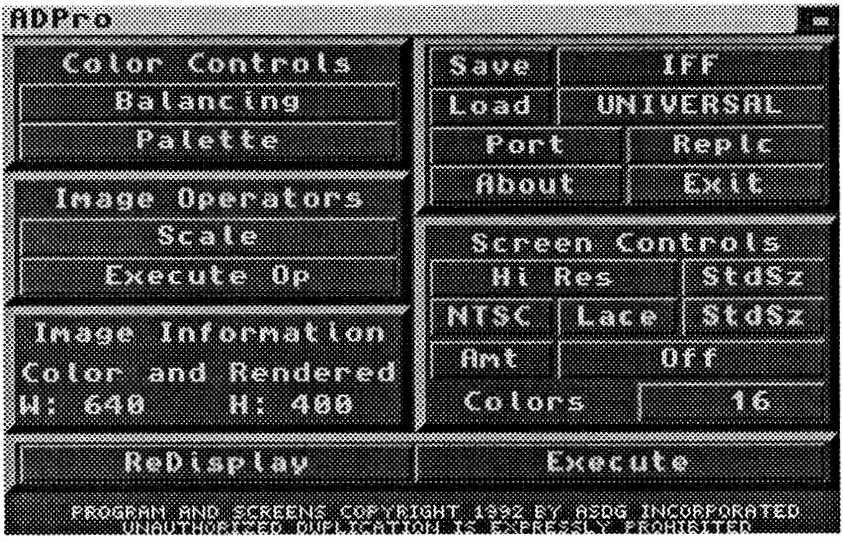


Figure 3.1: The main ADPro control screen.

3.1.2 Load

Aspects of the **Load** function are described more fully in Section 2.3. To summarize that information, the **Load** function in ADPro is actually implemented in separate programs called Loaders.

Each file to be loaded is searched for a color map. If a color map is contained in the file, it will be loaded along with the image data, provided that the ADPro palette is not in a **Locked** state. If the palette is **Locked**, ADPro will ultimately ask you if you really do wish to load the palette from the file.

If found, the palette will be inspected to see if it contains only shades of gray. If it does, the file's data will be expanded out into an 8 bit-plane grayscale image. If not, the file will be expanded out into a 24 bit-plane color image.

Depending upon the state of the **Orientation** button, the image may be rotated 270 degrees while it is being read from disk.

The expansion into 8 or 24 bit-planes is usually performed after the file has been loaded. However, if you are loading an Amiga displayable file and there isn't enough memory to hold the raw image data but there is enough memory to hold the displayable data, then it will be loaded without the conversion to 8 or 24 bit-planes. In this case, the image can be displayed, but none of the processing commands which depend on having raw data around will function.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

When loading files which are already in an Amiga displayable format, the image is directly viewable immediately after loading by selecting the **ReDisplay** button.

Depending on the load format, aborting a load while in progress may or may not produce displayable image data.

3.1.3 Save Format

The **Save Format** button displays the format in which the current image data will be saved the next time the **Save** button is pressed. For a listing of available formats, please see Chapter 5.

3.1.4 Save

The **Save** button allows you to save the current raw or rendered image data in the current **Save Format**. Please refer to Section 2.5 for general information about how to save images with ADPro. Please refer to the Chapter 5 for information about a particular saver.

3.1.5 Orientation

The Orientation button is described more fully in Section 2.3.2. To summarize that information, the Orientation button toggles between a portrait (**Port**) and landscape (**Land**) setting. The state of this button influences the orientation of the next image to be loaded. Using this feature in conjunction with the Horizontal_Flip and Vertical_Flip operators, you can produce 90 degree rotations through 0, 90, 180, 270 degrees.

3.1.6 Compositing

The Compositing button describes whether image compositing is enabled or not. This button toggles between **Replc** (the next loaded image will replace the current image in memory) and **Comp** (the next loaded image will be composited with the current image).

Please see Sections 2.3.3 and 2.3.4 for more information about image compositing.

3.1.7 About

Selecting the **About** button will cause several information panels to appear. The information displayed includes the revision and serial number of your ADPro. It will also tell you the number of bytes which ADPro is currently using as its primary imaging buffer. Finally, it asks for your cooperation in keeping Art Department Professional professionally supported.

3.1.8 Exit

Selecting the **Exit** button will cause ADPro to exit.

If the "don't save settings" flag is not enabled (see Sections 10.1 for a description of this), then most of the current program settings will be saved to a file called `ADProDefaults`. The next time you start ADPro (with the "use settings file" flag enabled), ADPro will try to find this file in the current directory. If it finds it, it will use the settings defined in that file to initialize the program. This allows you to retain many of the control screen and panel settings from session to session.

3.2 Image Operators

This section describes the command buttons available in the Image Operators area (located above the Image Information area) of ADPro's main screen.

3.2.1 Operator Format

Selecting the **Operator Format** button will either display the Operator chooser list (if you are running under AmigaOS 2.04 or later) or switch to the previous or next operator module in the list of operators.

For a listing of available formats, please see Chapter 6.

3.2.2 Execute Op

The **Execute Op** button performs the currently selected operator (shown on the **Operator Format** button).

For a description of what each operator will display, please see the respective sections in Chapter 6.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

3.3 Display Controls

This section describes the command buttons available in the Display Controls area (located at the bottom) of ADPro's main screen.

3.3.1 ReDisplay

Selecting the **ReDisplay** button will cause any Amiga compatible rendered image data to be displayed. No image will be displayed if there is no rendered image data available or the available rendered image data is not in an Amiga compatible format.

If you have made any changes to the color or screen controls and have not hit the **Execute** button, the image displayed after selecting **ReDisplay** will not show the effect of any of these changes.

3.3.2 Execute

Select the **Execute** button to cause any changes you might have made in the screen, color, or image controls areas to be put into effect.

If the only change you've made is a change of screen width or height, selecting **Execute** will usually cause an image to render instantaneously.

If the image you are rendering is taller than the currently defined screen, then the image will appear to vertically wrap from bottom to top while rendering. This is normal and is ADPro's way of keeping you informed as to its rendering progress. Clicking on the image screen while ADPro is rendering will cause the ADPro control screen to pop to front.

When an image is being displayed, and it is larger than the currently defined screen, you can scroll about the image using the four cursor keys located on your keyboard. Scrolling can be performed in smaller increments by depressing the `Ctrl` (Control) key in conjunction with one of the cursor keys.

3.4 Color Controls

This section describes the functions located in the Color Controls area of ADPro. Each of the functions located in this area contributes to or affects the choice of colors which will be used in the rendering calculation.

Figure 2.9 depicts how these controls fit in to the flow of data through ADPro.

Notice that the color controls merely filter but do not actually modify the raw data. Therefore, you can always undo a change made to a color control and get back to the original rendered image.

3.4.1 Balancing

Whenever ADPro renders an image, it passes all of the raw color or grayscale data through a series of adjustments before actually using them in the rendering calculation. Before discussing the color balancing features of ADPro, let's first give some background information about how the adjustments are applied.

ADPro stores each pixel of a color image as 3 values (one each for red, green and blue) each of which can range from 0 to 255. Grayscale pixels are stored as a single value which can range from 0 to 255.

Figure 3.2 shows a linear (neutral) color map. A color map is a relationship between input intensities and output intensities. Internally, ADPro maintains a color map for each of the red, green, and blue components of colored data, and a grayscale map for gray data.

The color map shown in Figure 3.2 is said to be linear, or neutral, because it is a single straight line going from corner to corner in the color map. Notice that an incoming intensity of 128 is output unchanged as 128. The same is true for all intensities from 0 to 255; that is, they are output unchanged.

Brightness

The brightness adjustment globally modifies the general brightness of an image. It does this by uniformly shifting the color map upwards or downwards. This is shown in Figure 3.3. Here, the two input intensities are 128 and 160 (a difference of 32 intensity levels). Notice that they are output as 192 and 224 (with exactly the same difference in intensity levels). Similarly, all input intensities will be shifted upwards (or made brighter) by the color map shown in Figure 3.3.

The brightness adjustment is not without its drawbacks. Notice that an input value of 0 (in the color map shown in Figure 3.3) is output as 64. This means that the darkest intensity in the image will have an intensity of at least 64 which may not be acceptable. Also, note that all values from 192 to 255 all map to the same value—255. This means all details which had intensity levels in that range will become lost.

The brightness control in ADPro ranges from -50 to 50, with 0 being the neutral value. Setting the brightness control to a positive value uniformly

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

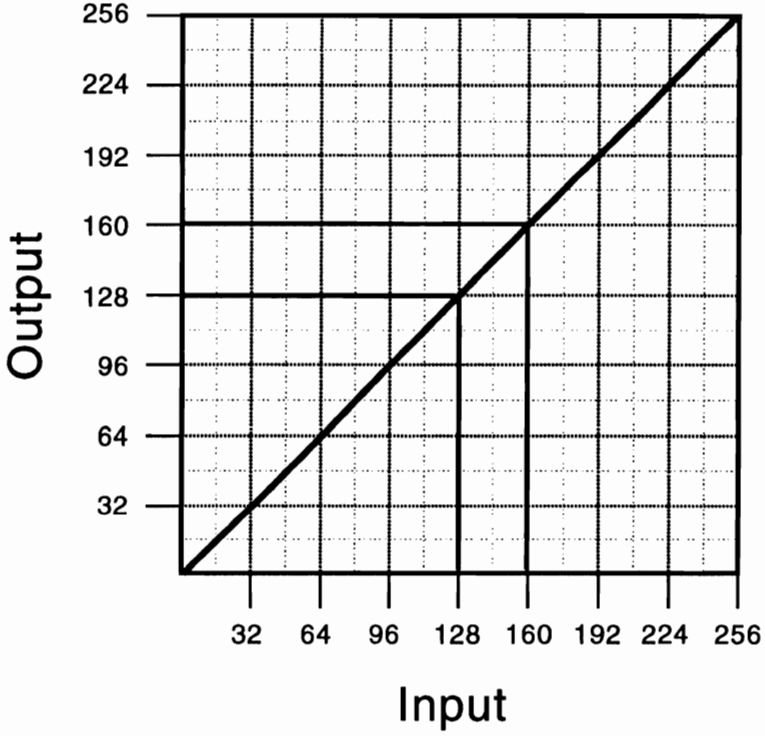


Figure 3.2: A linear (neutral) color map.

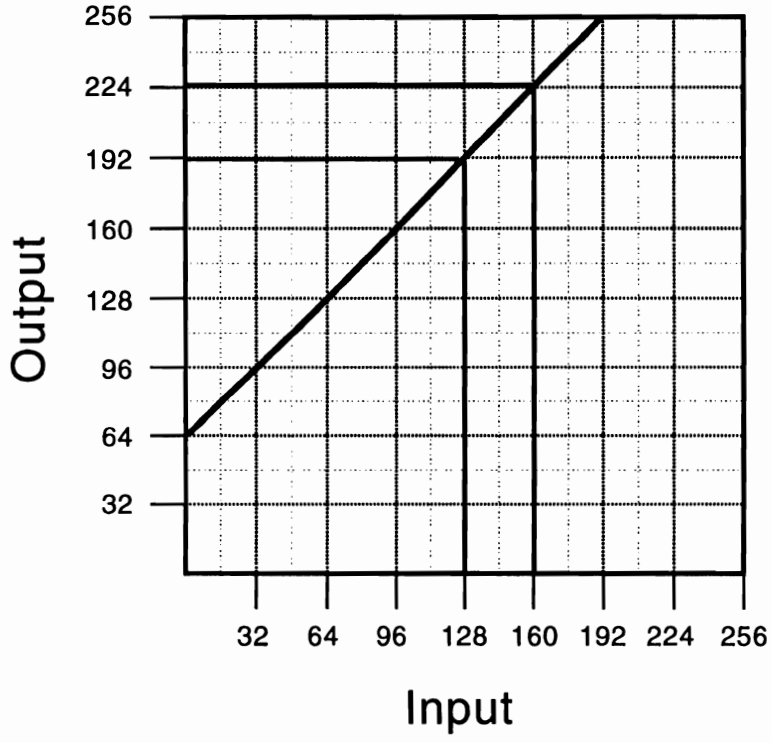


Figure 3.3: A color map showing an increase in brightness.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

shifts the color map upward (towards a brighter image). Similarly, a negative value causes the image to be shifted towards darkness.

Contrast

The contrast control globally modifies the contrast of an image. Contrast adjustments can be visualized by thinking of the neutral color map being pivoted around its center point. At one extreme, the color map becomes flat which means that all input intensities map to the same output intensity (no contrast). The other extreme is a vertical line for a color map. This produces an image with exactly two intensities (maximum contrast).

Figure 3.4 shows how input intensities 128 and 160 (a difference of 32 intensity levels) are mapped to output intensities 128 and 144 (a difference of only 16 intensity levels). Notice that the difference between two input intensities has been reduced. This produces a commensurate decrease in visible contrast.

Notice, again, that contrast loses some amount of visual detail just as the brightness adjustment. Specifically, you note that in the color map shown in Figure 3.4, input intensities which could have ranged from 0 to 255 are now restricted to the range of 64 to 192. This may or may not be acceptable for any given image.

The contrast control in ADPro ranges from -50 to 50, with 0 being the neutral value. Setting the contrast control to a positive value uniformly pivots the color map around its center in counter-clockwise direction (towards the vertical) which increases visible contrast.

Gamma

The gamma adjustment provides a way to significantly brighten an image without losing much detail. It does this by introducing a curve into the color map whereby the color map is shifted upwards or downwards (made brighter or darker respectively) but no portion of the color map gets clipped to the maximum or minimum values.

Figure 3.5 shows an example color map which has a positive gamma adjustment. Notice that the two input intensities, 128 and 160 are output as intensities 192 and 216. They are, therefore, brighter.

The gamma adjustment also affects the contrast of the image. In the darker part of the spectrum, contrast is increased. However, in the lighter part of the spectrum, contrast is decreased.

The gamma control in ADPro ranges from -50 to 50, where 0 represents no gamma adjustment.

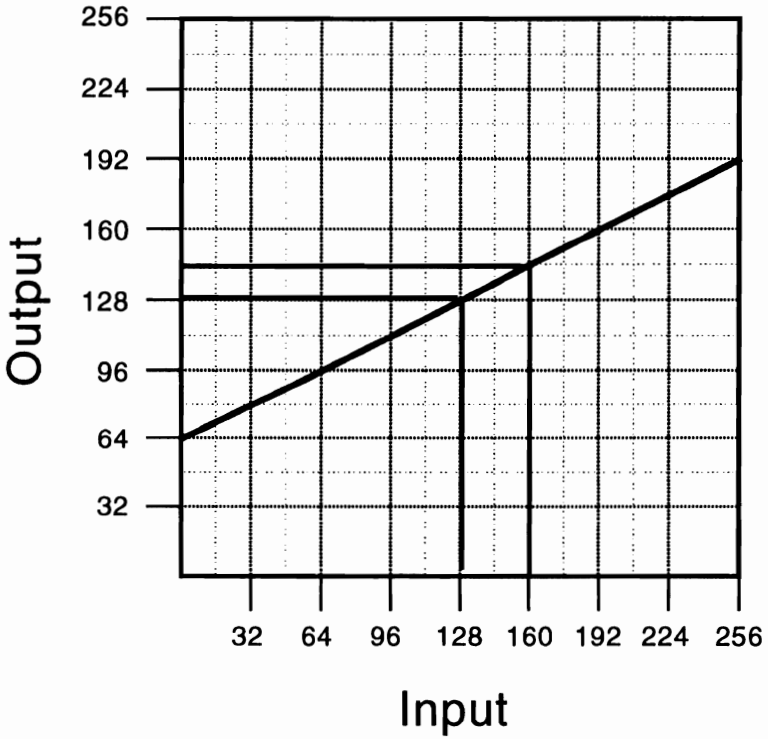


Figure 3.4: A color map showing a decrease in contrast.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

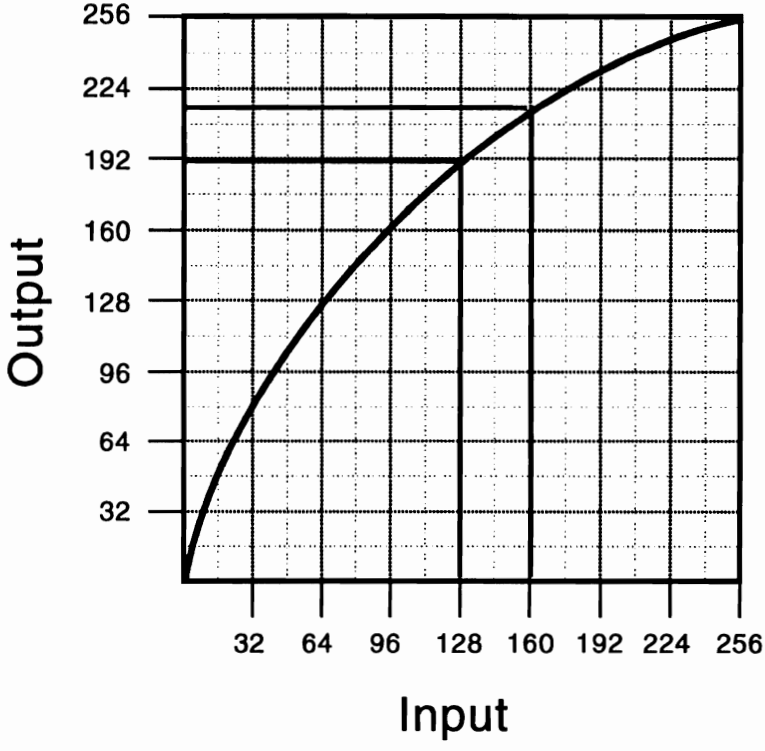


Figure 3.5: A color map showing an example of gamma correction.

The overall effect of gamma adjustment is usually quite satisfactory and we recommend its liberal use.

Red, Green, and Blue Adjustments

In addition to the previously described methods of color adjustment, ADPro also offers control over the individual brightness of the red, green, and blue data. Grayscale brightness is directly controlled by the main brightness control so no additional control is necessary.

Having this individual control allows for simple global color balancing changes. For example, an image which has far too much red can be balanced by decreasing the global brightness of all of the red data in the image.

Note that the individual brightness controls suffer from the same drawback as

Figure 3.6: The Balancing control panel.

the main brightness control. That is, the more heavily they are used, the more detail will be lost due to the color map being clipped against its minimum and maximum values.

The red, green, and blue adjustments in ADPro all range from -50 to 50, with 0 being the neutral value.

About the Balancing Control Panel

The panel which appears when you select the **Balancing** button (shown in Figure 3.6) contains the 6 color adjustments described above. Each is laid out as a horizontal slider.

At the bottom of the panel you'll find three buttons which will either accept, reset, or cancel any of the changes you've made to the color controls. Additionally, depressing either **Shift + Return** or **Alt + Return** will also accept any new values entered into the control panel.

Loading and Saving Balance Settings

Whenever raw image data is saved in the IFF format, ADPro includes the current locations of each of the balance settings. The positions of each balance

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

setting will be restored whenever an 8 or 24 bit-plane IFF file is loaded which contains stored settings.

The saving and restoring of the positions of the balance settings are not supported in any other format other than the IFF format. However, the effect of the balance settings will be preserved whenever raw image data is saved, regardless of format.

After loading a raw image (an image which is not color mapped), you may be presented with a panel which states that the "current balance settings do not correspond to actual data" when attempting to enter the Balancing control panel. This means that the image which was loaded did not contain ADPro-specific information about where to place the knobs in the Balancing control panel.

If the loaded data contained a color look-up table, that color look-up table will be loaded and effective until balance settings are "accepted". So, should you wish to use the color look-up table actually contained in a raw image, do not perform an **Accept** in the Balancing control panel. If you should enter the Balancing control panel, depress **Cancel** to preserve the color look-up table loaded from the file.

Accepting a set of balancing controls has the effect of completely overwriting any color look-up table which might have been loaded with image data.

3.4.2 Palette

ADPro provides very flexible and powerful palette controls. The Palette control panel is shown in Figure 3.7 and is described in this section. We'd like to mention that because the palette controls are so flexible and powerful, we cannot envision all the different effects and uses they may have. This section will describe the basic operation of the palette controls. We encourage you to experiment with the palette controls to discover new uses.

When are the Palette Controls Used?

In general, the palette controls are used in the rendering calculation only when the **Colors** button (one of the **Screen Controls**) is set to **CUST** (short for Custom).

If this button is not set to **CUST**, changes to the palette controls will not be honored, except for the palette's status and depth.

As described in the next section, a palette's status can be **Locked** or **Unlocked**. When **Locked**, the palette is "write-protected", which means that

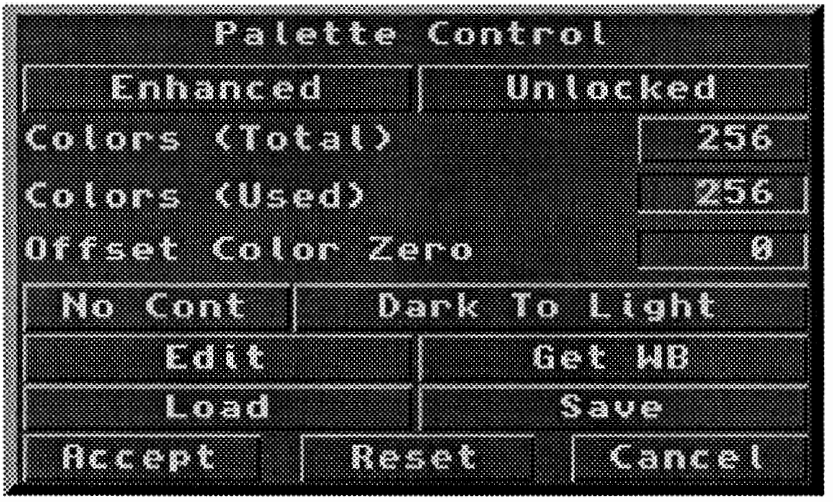


Figure 3.7: The Palette control panel.

ADPro's own color picking technology is disabled. The effect of a **Locked** palette is in force regardless of the setting of the **Colors** button.

As described in Section 3.4.2, when ADPro selects colors, it can do so at two levels of accuracy. For typical applications, the **Normal** accuracy is sufficient. However, some applications, such as creating imagery for display on non-Amiga computers, may benefit from **Enhanced** accuracy mode. The palette depth selection is in force any time ADPro selects a color on its own.

Palette Status

The palette status button can be set to one of two values: **Locked** and **Unlocked**. This defines whether or not ADPro will execute its color choosing routines prior to rendering or skip directly to rendering using the pre-existing palette contents.

When the palette is **Unlocked**, ADPro will analyze the raw image data and catalogue various statistics about the colors it contains. It will then optimally choose a set of colors which best approximates the entire image.

When the palette is **Locked**, ADPro will use the prior contents of the palette and will not choose a new one. Each pixel in the image will be rendered in the color chosen from the palette which best matches the intended color.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Referring to Figure 2.9, you can see that color adjustments and dithering still affect the final image even when the palette is **Locked**.

The palette **Locked** condition can also affect image loading. If the palette is in the **Locked** state and the image you wish to load contains a palette, ADPro will ask you if you wish to load the palette from the file or keep the previously defined palette. This will occur whether or not the **Colors** button is set to **CUST**.

Palette Accuracy

When ADPro selects colors, it can do so at two levels of accuracy. Typically, the **Normal** palette accuracy is sufficient for nearly all imagery intended for use on an Amiga based computer.

Other computer systems or display technologies can take advantage of more accurate color palettes. For example, VGA systems can display up to 256 colors simultaneously chosen from an 18 bit-wide palette. Commodore's Hi-Res graphics card can display up to 256 colors chosen from a palette 24 bits wide.

Consider the difference between a palette's depth and its width (accuracy). Palette depth can be thought of as the number of colors which can be simultaneously displayed. In the case of VGA, this is 8 bits deep (256 colors). The width of the palette can be thought of as the precision with which each color can be chosen. In the case of VGA, this is 18 bits wide.

A 32 color image can be displayed on both the Amiga's own screen and on Commodore's A2410 Hi-Res graphics card. For display on the Amiga's own screen, the 32 colors are chosen from a total spectrum of 4096 choices. For display on Commodore's Hi-Res graphics card, the 32 colors can be chosen from a spectrum of 16.7 million choices.

ADPro's **Enhanced** palette technology carries 24 bit-plane accuracy through all color conversion computations. By offering selectable palette accuracy, ADPro makes it possible to prepare high quality imagery for nearly any computer display hardware including future Amiga display technologies, such as the Advanced Graphics Architecture.

The enhanced palette technology does require some additional memory. If you are in a memory limited environment, you can disable enhanced palette mode operations. If you can spare the memory, we encourage the usage of the enhanced palette mode for all picture generation.

We recommend this mode for any type of processing, except in memory limited environments because it does provide better results in almost all modes and its speed is now acceptable.

Colors (Total)

If you set the **Colors** button (on the main screen) to **CUST**, the **Colors (Total)** button takes over the function of the **Colors** button in deciding the format of the rendered image. The choices include: **2**, **4**, **8**, **16**, **32**, **64**, **128**, and **256** colors, as well as **EHB**, **HAM**, and **HAM8**.

The choice you make here determines how many colors can occupy the image's palette. Therefore, the **Colors (Total)** button directly affects the **Colors (Used)** and **Offset Color Zero** values.

Colors (Used)

The **Colors (Used)** input field allows you to enter a number ranging from 2 to the total number of colors chosen. Normally, the number of colors used would be set to the maximum number possible for a given screen format. However, there are many situations where it is necessary to render in fewer colors than the screen format will permit.

For example, Amiga based genlocks show full color video through regions of the screen which are set to color 0. If a bitmap were to be rendered including color 0 and then genlocked, bits of genlocked video would poke through the bitmap anywhere a pixel where color 0 would be found.

To create an image which will appear solid when used with a genlock, simply don't use color 0 anywhere in the image. This can be accomplished by instructing ADPro to use one fewer color than is available. Then, specify a value of 1 in the **Offset Color Zero** input field. This will instruct ADPro to start filling in colors starting at color 1 rather than at 0. The result will be an image which contains $n-1$ (where n is the total number of colors possible) colors starting at color 1.

As another example, suppose you are required to use a specific 16 color palette and have to render an image with 4 colors (but in 4 rather than 2 bit-planes). This sort of example is a common requirement for animators. To do this, load and lock the required palette. Select a total number of colors of 16 but a number of colors to be used as 4. Specify a value in **Offset Color Zero** which will offset the colors used to the first of the 4 colors you wish to use. Note that this requires the 4 colors you wish to use to be stored contiguously in the palette.

The number in the **Colors (Used)** input field cannot exceed the total number of colors. Also, the sum of **Colors (Used)** and **Offset Color Zero** cannot exceed the total number of colors.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Offset Color Zero

The value in the **Offset Color Zero** input field is interpreted in different ways at different times. For example:

- When rendering and the palette is **Unlocked**, **Offset Color Zero** defines where in the palette ADPro will begin choosing new colors. For example, if **Offset Color Zero** is set to 4, colors 0 through 3 will not be changed and ADPro will start choosing new colors beginning at color 4.
- When rendering and the palette is **Locked**, **Offset Color Zero** defines where in the palette ADPro begins fetching colors for the rendering calculation. For example, if you wanted to render a 16 color screen with only 15 colors, you can either render with the first 15 colors or the last 15 colors. Setting **Offset Color Zero** to 0 will render with the first 15 colors while setting **Offset Color Zero** to 1 will render with the last 15 colors.
- When loading a color palette, **Offset Color Zero** defines where in the palette ADPro will begin storing the loaded values. For example, to create a 16 color palette from two 8 color palettes, load the first 8 color palette with **Offset Color Zero** set to 0. Load the second 8 color palette with **Offset Color Zero** set to 8. Then, lock the palette, and you're ready to render.

The value in **Offset Color Zero** can be in the range of 0 to the total number of colors minus 2. Also, the sum of **Colors (Used)** and **Offset Color Zero** cannot exceed the total number of colors.

Sort Direction

When ADPro chooses colors and places them into the palette, it can sort them by increasing or decreasing brightness. This button is a toggle which controls the sort direction. Of particular note is that color 0 is the color which will be shown as a border around non-overscanned images. In general, you'd like this to be as dark as possible (**Darkest To Lightest**) so as not to be distracting. However, this can be overridden by selecting **Lightest To Darkest**.

Contrast or No Contrast

When ADPro selects colors, it orders them dark to light or light to dark. In doing so, it may place two very similar colors in color registers 0 and 1. This makes viewing and editing of the palette very difficult since most palette

requesters use color registers 0 and 1 as their background and highlight pens, respectively.

If you set the Palette Contrast button to **Cont**, then ADPro will examine color register 0 and place a contrasting color in register 1. This eliminates the difficulty in using most palette requesters.

When the Palette Contrast button is set to **No Cont**, ADPro will order its palette with no special provision for distinguishing color registers 0 and 1.

Edit

Selecting the **Edit** button will bring up one of two color requesters. These are described below.

After editing, the effect of the new palette is immediately viewable by selecting the **ReDisplay** button in the **Commands** area. However, the image which will be displayed will simply be the previously rendered image with a new set of colors. To rerender the image (that is, go through the entire data flow as shown in Figure 2.9), you can select the **Execute** button.

The 256 Color Palette Editor

If you have enough memory, you will enjoy the benefits of being able to edit all 256 colors on-screen at one time. The new Palette Editor is accessed via the button marked **Edit** on the Palette control panel. If you do not have enough memory available to run this 256 color Palette Editor or if this editor cannot be located on disk, you will be presented with a limited palette editor which will be described later in this section.

The first thing to note is that although the Palette Editor provides access to all 256 colors on screen at one time, it does not circumvent the Amiga's 12 bit palette width. Therefore, colors which are quite close together in 24 bit space will be shown as the same color in the palette editor. For example, the color (128, 128, 128) will be shown as the same color as (127, 128, 129). This is because most Amigas can display only 16 shades of each primary, whereas the editor allows you to edit 256 shades of each primary.

Also, hitting the `Return` key in the integer input fields does not advance you to the next integer input field. You can use the `Tab` to move forwards and `Shift + Tab` to move backwards.

The currently selected color (the one whose values are usually shown in the **RGB** and **HSV** sliders and input fields at the bottom of the screen) is shown with a yellow highlight.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

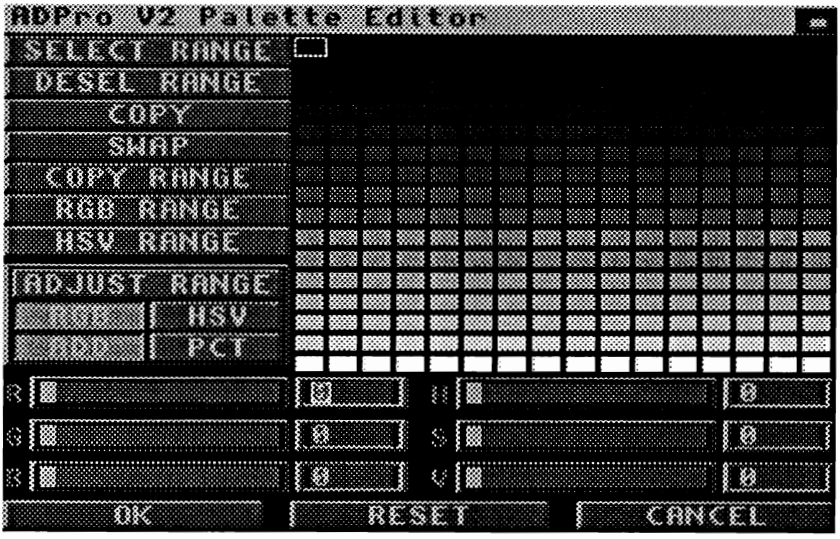


Figure 3.8: The 256 color Palette Editor.

The palette editor contains many powerful capabilities for operating on ranges of colors. Colors which are part of a range are shown with either green or red highlights.

The highlights themselves can sometimes alter your perception of the colors they surround. Therefore, you can press the right mouse button at any time to cause all graphical elements on screen to fade to black—leaving only the 256 colors. (In fact, you can use the sliders at the same time the screen is blacked out by first depressing the left mouse button over the slider, then depressing the right mouse button to black out the screen).

You can select a range in two ways. First, you can select the first element of the range, then hit the button marked **SELECT RANGE**, then select the last element in the range. All of the colors between the first and last elements will be highlighted in green.

If you hold down the **Alt** key and perform the steps above, the color will be highlighted in red. The difference between red and green highlighting will be explained later.

Range deselection can be accomplished the same way. Select the first element to be deselected, hit the **DESEL RANGE** button, then select the last element to be deselected. The first and last elements, and all colors in between, will be deselected.

You can copy any color from one location to another by selecting the color to be copied, then hit the **COPY** button, then select the color to be overwritten. Two colors can be swapped in the same way (instead of hitting the **COPY** button, hit the one marked **SWAP**).

Ranges which are defined with green or red highlights are said to be **explicit** ranges (as opposed to implicit ranges).

All of the commands with the word **RANGE** in their name operate slightly differently when an explicit range is defined or not. When an explicit range is defined, the command will affect only those colors which are part of the explicit range (highlighted by red or green). When an explicit range is **not** defined, the commands will affect all colors in between the first element selected and the second element selected.

The following example will demonstrate this:

1. Load in any color image and convert it to grayscale.
2. Select 256 colors (in the lower right corner of the screen) and hit **Execute**.
3. Enter the Palette Editor. Color register 0 will be selected yellow.
4. Hit the **SELECT RANGE** button. Notice that it stays highlighted.
5. Select color register 255 (bottom right corner). Notice all colors in between are highlighted with green.
6. Select color register 0 again (top left corner). Change this color to white using the sliders (simply move the **V** slider to 255).
7. Hit **COPY RANGE**. Notice it stays highlighted.
8. Select color register 255 (bottom right corner). Notice all colors in-between are changed to white.

Summary: *When an explicit range is defined, all members of the explicit range which are green are affected by a range command.*

1. Select a color register along the left edge of the color register grid, somewhere in the middle (of the grid's height).
2. Hit the **DESEL RANGE** button. Notice it stays highlighted.
3. Select the color register on the same line as the previously selected register, but along the right edge. Notice all of the registers in this line are no longer highlighted. They have been dropped from the explicit range.
4. Select color register 0 (top left corner). Change it to pure blue.
5. Hit **COPY RANGE**. Notice it stays highlighted.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

6. Select color register 255 (bottom right corner). Notice all colors in-between have changed to blue except for those not in the range, which were unaffected.

Summary: *When an explicit range is defined, colors which are not members of the range (even if they are in the middle of the range) are not affected by range commands.*

1. Hit the **DESEL RANGE** button. Notice it stays highlighted.
2. Select color register 0. All green highlighting should now be gone. You should have a color grid with all pure blue except for one line of pure white.
3. Select the first white color.
4. Hit **SELECT RANGE**.
5. Select the last white color. Notice all of the white colors are now highlighted with green.
6. Select color register 0 (top left corner). Make this color pure red.
7. Hit **COPY RANGE**.
8. Select color register 255 (bottom left). Notice that only the colors in the explicit range were affected.

Summary: *When an explicit range is defined and a portion of the range is contained between the end points of a range command, only those colors in the range are affected by range commands.*

1. Select the first highlighted color.
2. Hit the **DESEL RANGE** button.
3. Select the last highlighted color. No color should be highlighted with green at this point.
4. Select color register 0 (upper left corner) which should be pure red.
5. Hit **COPY RANGE**.
6. Select the color register still containing blue immediately before the first register containing red. Notice all color registers in between turn red.

Summary: *When no explicit range is defined, all range commands work on an implicit range defined by the selected color before the command button is hit...and the color selected after the command button is hit.*

If you select a color register or range of color registers with either **Alt** key held down, the register or range of registers will be highlighted with red instead of green.

Colors highlighted in red are part of the **explicit** range but their color values are locked. They will contribute a "place holder" to range commands, but their colors will not be affected by range commands.

This is best explained with the **RGB RANGE** and **HSV RANGE** commands and another example:

1. Ensure that no color registers are currently highlighted with red or green (i.e., no explicit range defined).
2. Select color register 0. If it isn't already, make it pure red.
3. Select color register 15 (top line, right corner). Make it pure blue.
4. Hit **RGB RANGE** (notice it stays selected).
5. Select color register 0. Notice that color registers 0 to 15 now contain a ramp from red to blue done in 16 steps.
6. Set colors 4 through 11 to pure white.
7. Define an explicit range of colors 0 to 15 by selecting 0, hitting the **SELECT RANGE** button, and selecting color 15.
8. Select color 4. Hit the **SELECT RANGE** button. Hold down the `Alt` key and select color 11. You should now have an explicit range defined from 0 to 15 with colors 4 through 11 being highlighted in red and the others highlighted in green.
9. Select color register 0. Hit **RGB RANGE**. Select color 15. *You should see nothing change.* This is because you computed a red to blue ramp with 16 steps, same as before...and it didn't affect the colors highlighted in red.
10. Select color register 4 (the first white color). Hit the **DESEL RANGE** button. Select color register 11 (the last white color). You should be left with 8 green highlighted colors separated into two groups of four, with 8 highlighted colors inbetween.
11. Select color register 0. Hit **RGB RANGE**. Select color register 15. Notice that the 8 white colors are unchanged but the 8 highlighted colors will change. They still contain a ramp from red to blue but the ramp is now done in 8 steps, not 16.

Summary: *Having colors in an explicit range highlighted with red means they contribute a place holder to range commands, but are not affected by them.*

The difference between **RGB RANGE** and **HSV RANGE** is that the **RGB RANGE** command builds a ramp through the RGB color space (while the **HSV RANGE** command uses the HSV space).

The **HSV RANGE** command is very handy for computing ranges in which all colors have the same intensity but have different hues, etc.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

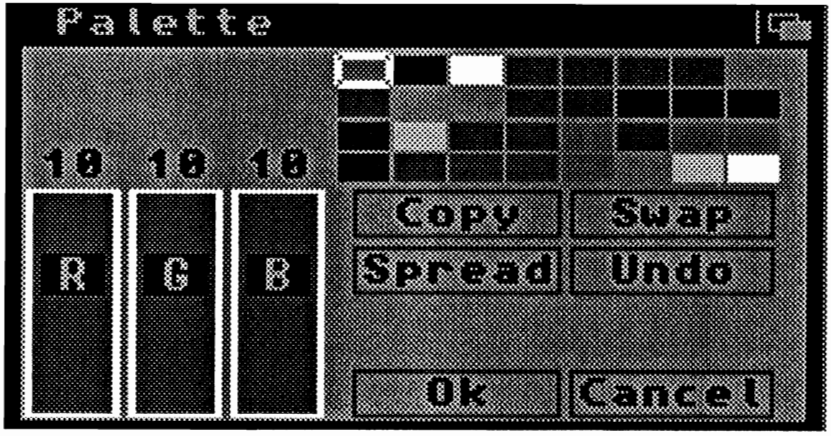


Figure 3.9: The limited Palette Editor.

The **ADJUST RANGE** command allows you to adjust the color balance of an entire range of colors simultaneously. To use the **ADJUST RANGE**, you must have an explicit range defined. Select which type of adjustment you wish to make, either in RGB space or HSV space and either ADDitive or PCT (percentage), then hit the **ADJUST RANGE** button. After you are done adjusting, hit the **ADJUST RANGE** button again to exit the adjusting mode.

The Limited Palette Editor

As mentioned earlier, if you don't have enough available memory to display this 256 color Palette Editor or if it cannot be found on disk, you will be given a limited palette editor to use.

You may manipulate one color at a time, chosen by clicking the left mouse button over one of the colored boxes at the right hand top of the color requester. These boxes represent the colors contained in each color register. The number of boxes displayed will correspond to the number of user definable colors in the screen mode you have selected with the **Colors** button.

The color which you have chosen to manipulate will be rendered in a rectangular area in the top left hand portion of the color requester.

Along the left hand edge of the color requester, three vertical sliders can be found which correspond to the red, green, and blue composition of the color being modified.

Below the color boxes (at the right middle of the color requester) are four

command buttons. These are defined below.

Selecting the **Copy** command will duplicate the current color into the next color register you click with the left mouse button. This will cause the two color registers to have the same value. The current color register will become the register you copied to.

Selecting the **Swap** command will exchange the contents of the current color register with the contents of the next color register you select with the left mouse button. The current color register will become the register you swapped to.

Selecting the **Undo** command will restore the previous state of the palette, undoing any previous **Swap**, **Copy**, or **Spread** command. The current color register is unchanged.

Selecting the **Spread** command will uniformly place colors between the current color register and the next selected register. The color registers you select become end points. Any color registers in between them are changed to a color in between the two end points.

Load

Selecting the **Load** button causes the file requester to appear. Using it, you can select a file from which ADPro will attempt to read a palette. All Amiga format images, such as 2, 4, 8, and 16 color images, have color palettes. However, raw image data, such as 18, 21 or 24 bit-plane files, do not. Palettes can also be loaded from IFF "brush" files as well.

Palette loading is affected by several other factors in the Palette control panel. Specifically, if the number of colors in the palette to be loaded exceeds the number of colors available to be loaded into, the excess colors will be ignored.

For example, if the total number of colors permissible at the time **Load** is selected (as defined by the value in the **Colors (Used)** input field) is 15, and the palette to be loaded contains 32 colors, only the first 15 will be loaded.

In fact, if the value in **Offset Color Zero** is non-zero at the time **Load** is selected, then the number of colors which will be loaded (in the example cited in the previous paragraph) will be lower than 16.

Note that a loaded palette will be overwritten if the palette is not in a **Locked** condition and you select the **Execute** button to render an image.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Save

The currently defined palette can be saved to a file by selecting the **Save** button. Upon doing so, the file requester will appear. After selecting a filename, the palette will be stored to disk.

Note that the saved file will contain only a palette. It will not contain any image data. Therefore, use caution when selecting a pre-existing file to store the palette, as that file will be overwritten with palette information. Its previous contents will be lost.

Get WB

This command is very useful for creating imagery which will be displayed on the Amiga Workbench. Selecting this command will cause ADPro to load the currently defined Workbench colors into the palette starting at the color defined by **Offset Color Zero**.

This command fetches *up to four colors* found in the Preferences structure. AmigaOS version 2.04 and later allow the Workbench to contain more than four colors. However, these colors are not found in the Preferences structure as defined by earlier versions of the operating system.

Therefore, consider the **Get WB** button as a short hand for loading only the first four colors of the Workbench. Should you require access to the palette of a more than four colored Workbench (possible under AmigaOS 2.04 and later), use the palette loading facility to load a palette from the `ENV:sys/palette.ilbm` palette preferences file or other pre-saved palette file.

3.5 Screen Controls

ADPro's screen controls offer a selection of 208 possible video modes, as many or more than any other Amiga program at this time. Once an image has been fully rendered, many of the changes from one screen mode to another are instantaneous.

NOTE Changes to any of the screen controls do not affect the raw or rendered data in any way. Therefore, such changes can be quickly undone.

Mode	Width
Low Res	320
Low Res/Overscan	368
Hi Res	640
Hi Res/Overscan	736
Super Hi-Res	1280
Super Hi-Res/Overscan	1472

Table 3.1: The horizontal resolutions supported by ADPro.

3.5.1 Horizontal Size

The various horizontal sizes are shown in Table 3.1. Note that some color modes preclude some horizontal sizes. These limitations are shown in Table 3.3.

Given a fully rendered image, changes from low resolution to high resolution will be instantaneous if the color mode chosen is allowed in both low and high resolution.

3.5.2 Vertical Size

The various vertical sizes are shown in Table 3.2. All color modes are supported in all vertical sizes. Selecting an NTSC vertical size on a PAL machine simply truncates the screen size at the NTSC lower boundary. Selecting a PAL vertical size on an NTSC machine simply extends the screen size below the visible lower border of your NTSC screen.

Given a fully rendered image, changes from one vertical size to another are instantaneous. Aspect correction for changing between interlaced and non-interlaced screens can be accomplished using the digital scaling capability of ADPro.

The Monitor Mode button (the leftmost button in the second row of buttons in the Screen Controls area) has been modified to support the new Super72 modes. The way you select the monitor has changed as well. Using a chooser list similar to the Dither list, you can select from the minimum set of monitors below:

DFLT
NTSC
PAL

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Mode	Height
NTSC	200
NTSC/Overscan	240
NTSC/Laced	400
NTSC/Laced/Overscan	480
PAL	256
PAL/Overscan	296
PAL/Laced	512
PAL/Laced/Overscan	592

Table 3.2: The vertical resolutions supported by ADPro.

and, if the Monitor icons are installed into your **DEVS:Monitors** directory (under AmigaOS 2.04 or later only), the following will show up as well:

VGA
SUP72

The Default (DFLT) mode will open the rendered image on a screen using the system defined (in the ScreenMode Preferences editor) Default monitor. If Mode Promotion is on (selectable in the IControls Preferences editor when running on AGA machines), then the Default setting will open up a screen in DblNTSC or DblPAL mode (depending upon which mode is currently in effect).

The Super72 (SUP72) mode will open a screen with the following characteristics:

width		height	
Low Res:	208	Non-Interlace:	300
Hi Res:	400	Interlace:	600
Super Hi Res:	800		

A new icon Tool Type called **SUPER72** is now supported. If listed in the ADPro's Tool Type list, the ADPro program will open up in Super72 (High Res Interlace) mode.

3.5.3 Super Hi-Res and VGA

If you are running on an Amiga equipped with the Enhanced Chip Set, then several new screen modes become available. These are:

- Super Hi-Res (1280), which allows double the horizontal resolution normally allowed by Hi-Res, including the Hi-Res overscan mode (1472).
- VGA which allows 480 or 960 lines vertically (depending upon the state of the interlace flag).

Be aware, however, that these modes have some significant restrictions (including the fact that they may contain only 4 colors chosen from a palette of only 64 choices). These modes are supported for completeness and future compatibility.

3.5.4 Dither

Dithering is a technique for achieving greater color fidelity at the expense of spatial fidelity (image sharpness). ADPro supports eight dithering types. These are: None, Floyd-Steinberg, Burkes, Sierra, Jarvis, Stucki, Random, Large Ordered, and Small Ordered dithers.

With the exception of the Random and Ordered dithers, the dithers are presented in the order of greatest to least effect on the image. However, counter to intuition, they range from fastest to slowest.

Floyd (1) (Floyd-Steinberg) is the tightest of the dithers and produces good results for all video modes. The **Stucki (5)** dither is the sparsest (and most time consuming to compute) and produces good results where just a little dithering is desired.

Dithers 1 through 5 are each "error diffusion" dithers. The added computation time of the higher numbered dithers results from incorporating a greater number of pixels into each dithering computation. This also accounts for why the higher numbered dithers affect the image less. That is, the error diffusion is spread over a larger number of pixels (thus affecting each pixel less).

The Random dither is useful in the preparation of successive images which will become part of an animation. While the other dithering techniques produce superior results, they may also produce an undesired flickering when multiple images are animated. The Random dither does not have this problem when animating.

The Ordered dithers (Lg Ord and Sm Ord) are also useful for animations. The large version uses a 16 x 16 dither matrix, while the small version uses a 4 x 4.

For the Random and Ordered dithers you can control how much of an effect they will have on the image. To the left of the **Dither Mode** button is the Dither Amount (shown as **Amt**) button. When pressed, a requester with a numeric input field will appear. You should enter a number between 1 and 256, with 1 having the least effect and 256 having the most.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If you are using a Dither Amount of 16 or less, then you should use the Small Ordered (Sm Ord) dither. However, if you're Dither Amount is greater than 16, use Large Ordered (Lg Ord).

After selecting the dithering style of your choice, and any other modifications to the image using ADPro's other image processing capabilities, select the **Execute** button (located in the lower right hand corner of the screen) to render the image.

The **Dither** button will activate the Dither Type chooser. See Table 10.2 for a listing of available dithers.

NOTE Dithering is applied during rendering and has no effect on the raw 8 bit-plane grayscale data or 24 bit-plane color data. Therefore, you may change and rechange the dithering setting at will without affecting the original data.

Dithering gives its best results when displayed in the high resolution video modes. Dithering provides a benefit in the HAM modes but can slightly increase the amount of HAM fringing present in the rendered image. Of the dithering choices available, the Random dither is the least likely to introduce HAM fringing but is the weakest dither.

3.5.5 Colors

The various choices of color mode are given in Table 3.3. This table also indicates the restrictions that the choice of color mode places on the horizontal size.

The color mode is selected using the **Colors** button. Clicking on this button will display the Number of Colors chooser list. See Section 2.1.3 for more information about the general use of chooser lists.

Notice that some entries in Table 3.3 indicate that that particular mode will not work in either low or high resolution Amiga screen modes. This indicates a number of colors which is not supported by the display capabilities of most Amigas. When rendering an image in 64 to 256 colors, no image will be displayable on the Amiga's screen (unless you are running the program on an AGA-equipped Amiga). Rendered image data is still available, however, for display on non-standard display devices such as the Commodore A2410 Hi-Res graphics card or for saving using a Saver which supports that particular number of colors.

For a detailed description of the **CUST** color mode setting, please see Sections 3.4.2 and 3.4.2.

Setting	Colors	Bitplanes	Low Res	Hi Res
2	2	1	Y	Y
4	4	2	Y	Y
8	8	3	Y	Y
16	16	4	Y	Y
32	32	5	Y	A
64	64	6	A	A
128	128	7	A	A
256	256	8	A	A
EHB	64	6	Y	A
HAM	4096	6	Y	A
HAM8	262144	8	A	A

Table 3.3: Color modes and other information. The mixtures of color and display modes marked "A" are possible but require non-standard hardware or an AGA-equipped machine. ADPro allows these modes to be computed but will display them only if the appropriate enhanced hardware is available.

3.5.6 A-RES and A-HAM Restrictions

The A-HAM data format is a variation on Sliced HAM (SHAM) popularized by Rhett Anderson, formerly of Compute Magazine. In A-HAM, a new set of 16 base color registers is chosen for each line in the image. This increases the color fidelity of a HAM image at the expense of increased machine and display overhead.

ADPro will read A-HAM images in the same file format (as registered with CATS, Commodore Applications and Technical Support) used by NewTek's Dynamic HAM image files.

The A-RES modes are 16 color hi-res variants developed by ASDG, Incorporated. The A-RES file format was designed to be backwards compatible with NewTek's Dynamic Hi-Res format as registered with CATS.

A-RES allows all 4096 colors supported by the Amiga to be present on one hi-res screen at the expense of extreme processor overhead while an A-RES image is being displayed.

NOTE The A-RES modes are not for general purpose use as are the standard Amiga display modes. There are many limitations to what can be done while displaying an A-RES image and not all images lend themselves to rendering in this mode.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The generation of A-RES and A-HAM images is not fully supported. While ADPro will continue to be able to read these images, it can no longer generate them. This decision was reached after careful consideration based upon knowledge of future developments.

3.5.7 Other Overscan Sizes

Tables 3.1 and 3.2 provide the dimensions of the overscan screens directly supported by ADPro. These values were suggested by the hardware documentation provided by Commodore. However, the Amiga can produce horizontal and vertical overscans larger than the values directly supported by ADPro.

If you wish to produce an image conforming to a different overscan size, simply render the image to the desired size (since ADPro will render at any size) and use a program other than ADPro for display purposes.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 4

Loaders

This chapter provides detailed descriptions of the use and operation of all loaders which are included with ADPro. ARexx control of these loaders is described in Chapter 7.

4.1 ALPHA

The ALPHA loader is an IFF-ILBM loader that allows you to use a grayscale IFF-ILBM image as an alpha channel, controlling the transparency of the image to be loaded and composited over the current image in ADPro's image buffer.

Currently, the file specified as the alpha channel must be a grayscale IFF and must have the same width and height as the image being loaded. In this alpha channel file, the whiter the pixel the more the foreground (loaded image's) pixel will replace the background (current image's) pixel, and the darker the pixel the less it will replace it. You can create alpha channel files in any paint program for use as sophisticated masks or artistic effects.

To use the ALPHA loader from ADPro's control screen, follow the steps below:

1. Load the background image, if it isn't already loaded, into ADPro's image buffer.
2. Set the compositing button on ADPro's main control screen to **Comp**. If this button says **Replc**, click on it once to toggle to compositing mode.
3. Select the ALPHA loader from the Loader list and press the **Load** button. This step launches the ALPHA loader.

4. From the file requester that is displayed, select the image to be composited (known also as the foreground image). Notice that the file requester's title bar displays the type of file being requested.
5. The Image Compositing control panel will appear. Press the **OK** button to proceed with this operation or **Cancel** to abort it. Additionally, you can press **No Comp** to replace the background image with the selected image, thereby cancelling the operation. You can also change the offset and mix values if you wish.
6. Another file requester will appear from which you should select the alpha channel file. Notice once again how the file requester's title bar displays what is needed.

   Be sure that the file you select is a grayscale image (in IFF-ILBM format) with the same width and height as the composited (foreground) image, or else the mix levels defined in the Compositing control panel will be used (instead of those defined by the alpha channel file).

After you select a proper file, the foreground image will be composited over the background image.

Be sure you do not delete the `.ALPHA` and `ALPHA` files from the `Loaders2` directory. The ALPHA loader needs both of these files to operate. These files are placed in this directory by the installation program.

4.2 ANIM

The ANIM loader allows you to load any frame within a properly defined IFF-ANIM (Op Mode 5) animation file.

After selecting the ANIM loader and pressing the **Load** button, a file requester will appear for you to select an ANIM file. Once a valid IFF-ANIM file has be selected, the ANIM loader control panel will appear, as shown in Figure 4.1. The loader will then ask you for the specific frame number (shown in the **Frame Number:** input field) to load. Assuming that the animation has that frame number, it will load that frame into ADPro.

The next time you load an ANIM frame, the number specified in the **Frame Number:** input field will be incremented, assuming you will want to load the next frame in the animation.

Just above the **Frame Number:** input field is the **Number of Frames:** indicator. It will display **0** until you press the **Count Frames** button, which will count how many frames are in the ANIM file and display this value.

A time-saving feature of the ANIM loader is that it remembers the position of the last frame you selected to load. This allows you to load successive

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

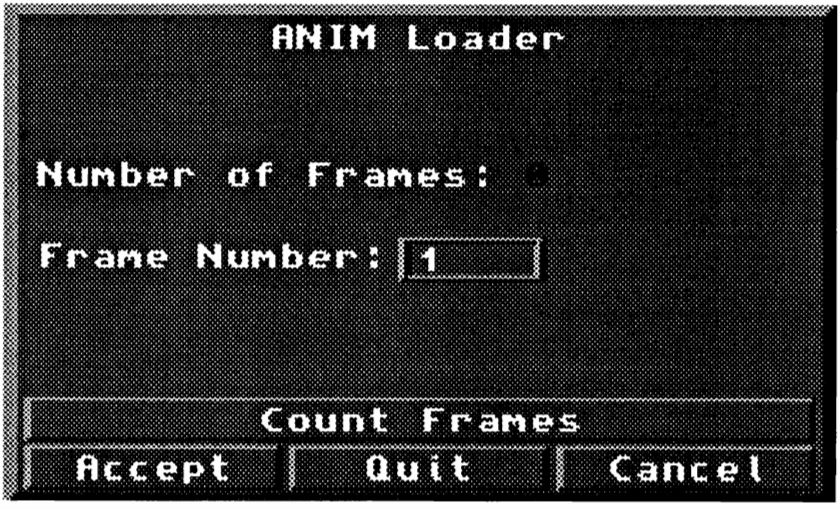


Figure 4.1: The ANIM loader control panel.

frames without having to wait for the loader to search for that frame. For animations made up of a few frames, this feature might not be as evident as with a "multi-frame" animation.

To exit the loader without actually loading a frame, select **Cancel**. The control panel will disappear, returning control back to the main screen. If you want to use the time-saving feature just described, select this button to exit the control panel.

If you want to exit the control panel without saving the information about the last frame loaded, select the **Quit** button. **Quit** will close the ANIM file, allowing you to use it in other programs.

NOTE If you want to play the animation with an ANIM viewer, you MUST select the **Quit** button after finishing all of your work (or exit the ADPro program). If you fail to do this step, the ANIM viewer will not be able to read the file.

4.3 BACKDROP

Don't let the name, BACKDROP, fool you. The BACKDROP loader is one of the most important loaders provided with ADPro. Using it, you can create a "canvas" upon which you can add other images using ADPro's image compositing features.

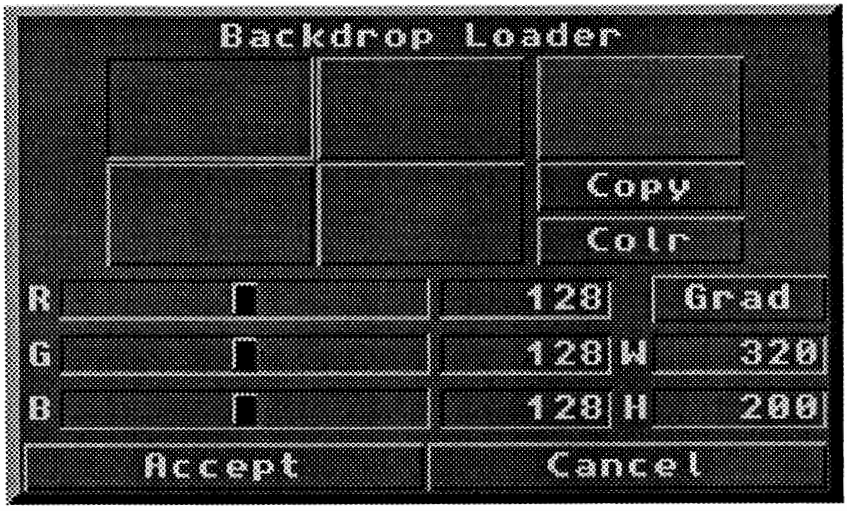


Figure 4.2: The BACKDROP loadercontrol panel.

At the same time the BACKDROP loader creates an empty bit-map of some given size, you can ask it to fill the bit-map with a specific color or even a gradation of colors. Gradated background fills are very common in presentation and business graphics.

Figure 4.2 shows the user interface for the BACKDROP loader.

The BACKDROP loader can prepare one of two different types of bit-maps. A 24 bit-plane raw color bit-map can be prepared by selecting a setting of **Colr**, which is shown in Figure 4.2. Selecting this button causes it to toggle to its alternate state, **Gray**, which instructs the BACKDROP loader to prepare a 8 bit-plane gray scale bit-map.

The size of the bit-map to be prepared is specified in the input fields to the right of the markings **W** (for bit-map width) and **H** (for bit-map height).

The rocker switch above the width and height fields determines how the resulting bit-map will be filled. If in the **Grad** setting as in Figure 4.2, the bit-map will be filled with a smooth shaded gradated fill. If set to **Fill** the bit-map will be set to one constant color.

You may specify the nature of a gradated fill by selecting each of the four blank rectangular buttons in the upper half of the control panel. Each of these buttons represents the corresponding corner of the image. That is, the upper left hand button corresponds to the upper left hand corner of the image.

These buttons are enabled whenever you have a gradated fill selected. If you

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

were to toggle the fill type rocker switch to **Fill**, these buttons become disabled.

A non-gradated color fill can be specified simply by setting the desired color into the red, green, and blue sliders or input fields. When a grayscale bit-map is selected, the red and blue sliders/fields disappear leaving only a slider/field pair marked **G** (for gray).

As you change the red, green, or blue (or just the gray) component of the color being entered, the numerical value of the color will be updated. The corresponding color will be displayed in a small rectangular area above the button marked **Copy**. Remember that you are entering color components with 24 bit accuracy. The color displayed above the **Copy** button is only accurate to the 12 bit limit of the Amiga's palette. Therefore, slight adjustments in the numerical value of the color may not be visible on-screen.

When ADPro constructs a gradated fill, the color of each pixel is computed as a linear interpolation both horizontally and vertically.

To copy the color of one corner to another, select the corner you wish to copy from, then select the **Copy** button, and, finally, select the corner you wish to copy to.

4.4 BACKLINE

The BACKLINE loader is similar to the BACKDROP loader in that it is used to create backdrops. BACKLINE differs from BACKDROP in the kinds of gradated background that it can produce. BACKDROP allows you to specify the color of the four corners of the backdrop. This allows for nice vertical or horizontal sweeps. With creative use, the BACKDROP loader can even be coerced into producing diagonal fills.

The BACKLINE loader allows you to specify three colors which will be used to create the fill. BACKLINE creates a smoothly shaded ramp from the first color (the **Start** color) to the second color (the **Middle** color), then it smoothly shades from the **Middle** color to the third color (the **End** color). The ramp itself can be oriented in any one of eight directions (horizontally, vertically, or along the diagonals).

The shading of the ramp is designed to make it appear like a curved surface. This is done by using the **Start**, **Middle**, and **End** colors to define a mathematical entity called a "spline."

Figure 4.3 shows the control panel for the BACKLINE loader. You can control the direction of the ramp by selecting any one of the eight directions arranged in a square. In the center of the square, the BACKLINE loader will display the color which is currently being entered (via the **R**, **G**, and **B** sliders and

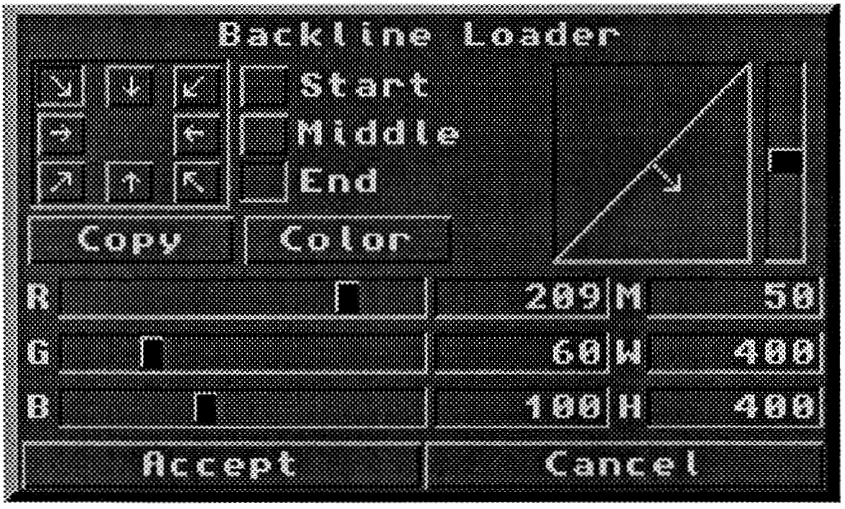


Figure 4.3: The BACKLINE loader control panel.

input fields). You can specify which of the **Start**, **Middle**, or **End** colors you can control from the **R**, **G**, and **B** sliders by selecting one of the three buttons with the same name.

At the top right of the control panel, you will find a square which in Figure 4.3 contains an arrow and a diagonal line. The arrow will always correspond to the ramp direction you selected and is an additional form of feedback confirming your selection. The diagonal line represents the location of the **Middle** color with respect to the **Start** and **End**. Using the slider (or the input field marked **M**), you can specify how (spatially) close the **Middle** color will be to the **Start** or **End**. The values you can specify here range from 0 to 100. A value of 0 means that the **Middle** color will coincide with the **Start** color, while a value of 50 means that the **Middle** color will lie halfway between the **Start** and **End** color, and a value of 100 means that the **Middle** color will coincide with the **End** color.

Note that if you specify a diagonal ramp with a **Middle** color position of 50 percent, the **Middle** color will, in fact, lie along the diagonal (from corner to corner) only if the area you are creating is truly square.

You can use the button marked **Copy** to duplicate a color from one place (**Start**, **Middle**, or **End**) to another. Select the color you wish to duplicate, select the **Copy** button, and finally select the destination. The source color will be copied to the color you specified as the destination.

The rocker switch marked **Color** allows you to specify whether the BACKLINE

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

loader should create a grayscale or color backdrop. The gadgets marked **W** and **H** allow you to specify the width and height of the backdrop.

Hitting the **Accept** button causes a backdrop to be created according to the values you specified. The **Cancel** button exits the loader without causing a backdrop to be created.

4.5 BMP

The BMP format is the image file standard under Microsoft Windows 3.0 (and higher), much like IFF is the standard on the Amiga. BMP files come in 1, 4, 8 and 24 bit-plane varieties. While BMP files may be compressed using RLE (run-length encoding), we have not found an application which takes advantage of this capability. Further, the design of BMP's compression with respect to 24 bit-plane images virtually assures that compressed images will be larger than the original versions.

ADPro's BMP loader will read both compressed and uncompressed BMP files in all four depths.

4.6 CLIPBOARD

The CLIPBOARD loader allows ADPro to read pictures from the Amiga's Clipboard device. The Clipboard is a resource which can be shared between applications which support it. Note that text as well as images can be placed into the Clipboard. The CLIPBOARD loader can make use of only picture type information.

Note that the Amiga's Clipboard device is not heavily supported, and those applications that do support the Clipboard may not be expecting any other type of image data to be in the Clipboard except standard Amiga displayable images. Also note that the Clipboard device is often assigned to **RAM:**. This means that any image data written to the Clipboard will consume memory.

As an example of using the CLIPBOARD loader and saver, the icon editor (named IconEdit) supplied with Workbench 2.0 (and later) can copy from and paste to the Clipboard.

To load an icon into ADPro, drop the icon on the icon editor's window and select the **Copy** menu item. The icon is then written to the Clipboard. In ADPro, select the CLIPBOARD loader (and load the image). Instantly, the icon shows up in ADPro's image memory.

To save an icon from ADPro, simply reverse the steps. Use the CLIPBOARD

saver to save your prepared 4 color image (see Section 3.4.2). From the icon editor, select the **Paste** menu item. Instantly, the image shows up in the icon editor's image memory.

4.7 DPIIE

The DPIIE format was defined by Electronic Arts for use in their product "Deluxe Paint II Enhanced" for the IBM P.C. DPIIE images contain 256 colors. Deluxe Paint II Enhanced for the P.C. saves images with fewer than 256 colors in the standard Amiga IFF format.

To read an image in the DPIIE format, select the DPIIE loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in the DPIIE format.

4.8 DV21

The DV21 loader can process the 21 bit-plane files saved by NewTek's Digi-View 3.0 **only**. A 21 bit-plane file saved by Digi-View 4.0 must be read by the IFF loader.

Files loaded by this module are always considered to be color images. As the image is loaded, it will be padded to a full 24 bit-planes.

To load a 21 bit-plane file saved by Digi-View 3.0, select the DV21 loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in the DV21 format.

4.9 FRAMEGRABBER

The FRAMEGRABBER loader allows direct control over Progressive Peripherals, Inc.'s FrameGrabber (a video digitizer). The FrameGrabber can digitize a low resolution interlaced image in 12 bit color in a single video frame time ($\frac{1}{30}$ of a second).

The FRAMEGRABBER control panel is shown in Figure 4.4. Selecting the **Cancel** button will exit the loader without digitizing an image. Any image previously loaded in memory will be unaffected. Selecting the button marked **Digitize** will cause a full video frame to be digitized at that moment.

If you have routed your Amiga video through the FrameGrabber (consult the documentation which comes with the FrameGrabber), selecting the button

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

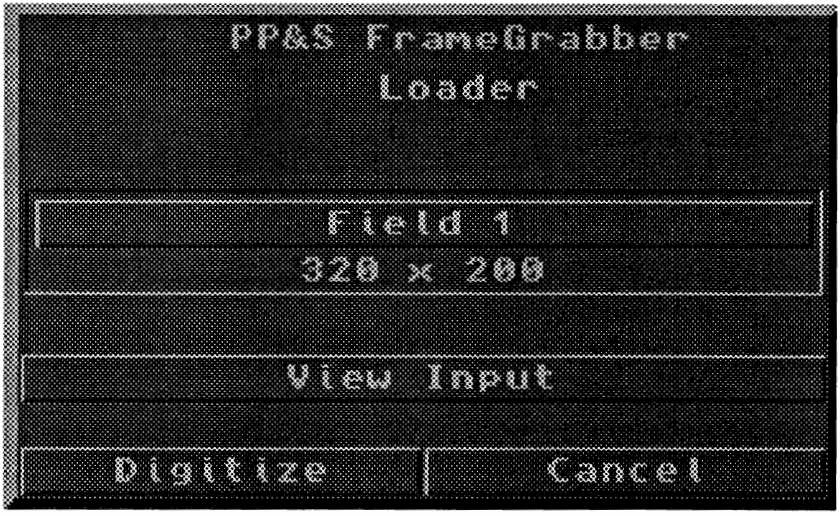


Figure 4.4: The FRAMEGRABBER loader control panel.

marked **View Input** will cause the Amiga display to be replaced with the live video coming through the FrameGrabber. All input and processing on the Amiga will be suspended until you click the left mouse button or press the **Tab** key or **Space Bar**. You may also depress the **c** key to cause the currently displayed video to be captured.

When the control panel is displayed, depressing the **Tab** key or **Space Bar** will toggle between your Amiga display and live video. This is equivalent to selecting the button marked **View Input**. As mentioned above, while displaying live video, all input and processing on your Amiga will be suspended until you press the left mouse button, or press the **c** key, the **Tab** key, or **Space Bar**.

The **c** key will also cause a grab to take place when the control panel is displayed. This is provided for consistency with the FrameGrabber's own software.

The image may be grabbed in one of four ways based upon the intrinsic capabilities of the FrameGrabber. The FrameGrabber physically grabs two 320 by 200 fields (each in $\frac{1}{60}$ of a second) in 12 bit color. You can choose to download either field alone, resulting in a 320 by 200 image. Or, you can download both fields stacked one upon the other, resulting in a 320 by 400 image. Or, finally, you can download both fields interlaced together, also resulting in a 320 by 400 pixel image.

If you are digitizing a still scene, we suggest using the interlaced field mode

(320 by 400 pixels) as it produces the image data which yields the most exact reproduction of which the FrameGrabber is capable.

If you are digitizing a moving scene, then you may wish to download only one field rather than both. This is because a single field is digitized in a 60th of a second and can stop motion better than two fields together (which may contain some motion blur). In fact, if you select the stacked field mode, then both fields will be downloaded such that you can choose which field to save. In the stacked mode, a 320 by 400 pixel image is produced from two 320 by 200 pixel images stacked one above the other. Using one of the crop operators, you can pick which field contains the better looking data.

NOTE The FRAMEGRABBER loader does not operate correctly on PAL machines.

4.10 GIF

The GIF format was defined by CompuServe Information Services, Inc. for storing images in a compressed form on their information network. The GIF format is very widely used on IBM and Apple computers. GIF images may contain from 1 to 256 colors.

The ADPro GIF loader supports both the 87a and 89a GIF standards, including interlaced GIF images. It supports multi-image GIF files as well.

To read an image in the GIF format, select the GIF loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in GIF format.

4.11 HAME

The HAME loader will read files formatted for use with the HAM-E display enhancer from Black Belt Systems. This file format encodes enhanced color information in a standard 16 color high resolution IFF file. The loader decodes this formatting information and converts the loaded image back into true color data.

The HAM-E format places one or more "cookie" lines in with the image data to control how the data is to be interpreted. These lines are stripped from the image as they are read. Therefore, it is normal and expected to load in an image and receive fewer lines than you were expecting.

Note that you can load HAM-E formatted images using either IFF or HAME loaders. If you use the IFF loader, then you can directly view the loaded

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

image on a HAM-E since the IFF loader does not disturb the embedded control information in the file. The HAME loader actually strips away the embedded control information after using it to convert the encoded image into raw 24 bit-plane color or 8 bit-plane gray data.

4.12 IFF

The IFF loader (also referred to as the Super-IFF loader) allows you to load any properly defined IFF-ILBM picture file, including all of the following:

- Commodore standard IFF files in 1 through 5 bit-planes, corresponding to 2 through 32 colors, respectively.
- Commodore standard IFF files in the 64 color Amiga Extra-Halfbrite (EHB) format.
- Commodore standard IFF files in the 4096 color Hold-And-Modify (HAM) format.
- IFF files in Sliced-HAM (SHAM) format (can be read but not written).
- IFF files in A-HAM or Dynamic HAM formats (4096 colors).
- IFF files in A-RES or Dynamic Hi-Res formats (4096 colors in hi-res).
- Commodore standard IFF files in 12, 15, 18, 21, and 24 bit-planes (4096, 32768, 262144, 2097152, and 16777216 colors, respectively). This includes 21 bit-plane files created by Digi-View 4.0.

The Super-IFF loader will look for a palette in the image file. If there is one, it will be tested to see if it contains only grayscales. If it does, the image will be loaded into 8 bit-planes and will be treated as grayscale image data. Otherwise, the image will be loaded into 24 bit-planes and treated as a color image.

When attempting to load an image already in a rendered format (such as the 1 to 6 bit-plane formats) and there isn't enough memory to convert the full image into 24 bit-planes or 8 bit-planes if grayscale), the Super-IFF loader will simply load the image and not convert it into a deep bit-plane format. This means you will be able to view the image, edit its palette, and even save the image back out in a file format supporting the same depth, but you will not be able to perform any other of ADPro's image processing functions.

4.13 IMPULSE

The IMPULSE formats are defined by Impulse, Inc. for use with their color products, such as Turbo-Silver and Imagine. ADPro's IMPULSE loader sup-

ports reading the RGBN and RGB8 IMPULSE formats. The RGBN format stores 4 bits per primary color, giving a total of 12 bit-planes. The RGB8 format stores 8 bits per primary, giving a total of 24 bit-planes of color information.

To read an image in either format, select the IMPULSE loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in either IMPULSE format.

Files in the IMPULSE format are always interpreted as color images.

4.14 IV24

The IV24 (Impact Vision 24) from Great Valley Products provides a 24 bit-plane framegrabber as well as being a 24 bit-plane display device. The IV24 loader provides control over the IV24's framegrabbing capability.

NOTE You must have an IV24 installed in your machine and you must have installed the `fye.library` in your `LIBS:` directory to make use of this loader. If these conditions are not met, selecting this loader will cause an error message to be displayed on ADPro's main screen.

The control panel shown in Figure 4.5 is presented when you execute the IV24 loader. You can use the **Width**, **Height**, and **Offset** controls to define a rectangle within which the image data will be grabbed.

The upper left hand corner of the IV24 is at an **X Offset** of 0 and **Y Offset** of 0.

Note that the **Width** plus the **X Offset** must be less than or equal to 768. The **Height** plus the **Y Offset** must be less than or equal to 566.

If your video set-up allows you to view the IV24 display on your Amiga monitor, selecting the **View** button will switch to IV24 video. Clicking the left mouse button while watching IV24 video brings you back to Amiga video.

Note that while watching IV24 video (as a result of pushing the button marked **View**) you can press the `d` key on your keyboard to cause a framegrab.

Selecting the **Digitize** button causes a framegrab to take place. Selecting the **Cancel** button exits the IV24 loader with no action taking place.

NOTE This loader does not support composition nor landscape style loading.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

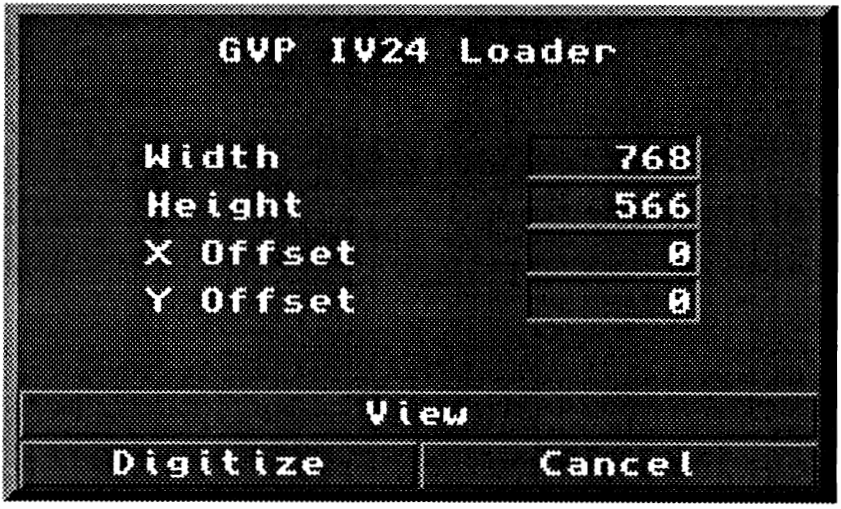


Figure 4.5: The IV24 loader control panel.

4.15 JPEG

The JPEG loader can read and decompress images written to the JFIF file format. The JFIF format is a standard for storing images compressed using the Joint Photographic Expert Group compression algorithms. JPEG can compress images to a fraction of their original size but does so with some loss to the image. That is, the decompressed image may vary slightly from the original image.

The JPEG loader control panel is shown in Figure 4.6. The only user control is whether or not smoothing is enabled. JPEG actually compresses small square regions of the image individually. Sometimes, when an image is very heavily compressed, you may see evidence of these small squares after decompression. Smoothing is a blurring operation which takes place during decompression and, therefore, is written so that it does not blur the entire image, but only the borders between the small squares. This can often mask or hide some of the artifacts which JPEG compression can introduce.

Pressing the **Accept** button will cause the file requester to appear. You can use it to select the file to be read.

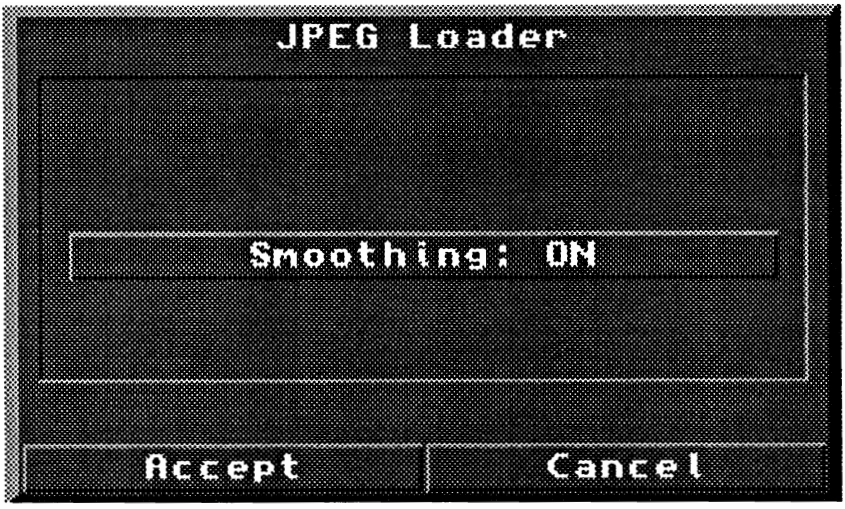


Figure 4.6: The JPEG loader control panel.

4.16 MACPAINT

The MACPAINT loader interprets files in the MacPaint format. This format provides for black and white (1 bit-plane) images only. Files are converted to 8 bit-plane gray scale as they are loaded.

4.17 PCX

The PCX format was defined by the ZSoft Corporation for use in their product, PC Paint Brush. PCX images may contain from 1 to 256 colors, including special palettes for IBM PC EGA and CGA display boards. PCX images may now also contain 24 bit-planes of color information.

The ADPro PCX loader will correctly interpret most PCX images including those with CGA and EGA palettes. However, some CGA images may contain misleading palette information which ADPro cannot help but follow.

To read an image in the PCX format, select the PCX loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in the PCX format.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

4.18 POINTER

The POINTER loader can be used to digitize any mouse pointer sprite which can be displayed while this loader is running. Those applications which do not multitask (such as some games) cannot have their mouse pointer's digitized since ADPro cannot run at the same time. Being able to digitize the mouse pointer adds realism to screen grabs performed with the SCREEN loader.

Selecting the **Load** button causes the POINTER loader's control panel to appear. At this time, a hot-key sequence is installed in the input stream of your Amiga. When you press the **Right Alt + Right Shift + Backspace** key sequence, the mouse pointer currently being displayed will be digitized.

If compositing mode is enabled, you will have an opportunity to place the newly digitized pointer over a predefined image already in ADPro's memory. Note that pointers are always digitized in low resolution (or high resolution, if running under Workbench 3.0 or later) pixels and ADPro has no way of knowing what size pixel you intend it to maintain within its memory. Therefore, some combinations of original screen resolutions can lead to a slightly distorted composition of the mouse pointer.

When compositing with the mouse pointer, its transparent color is automatically discarded to allow the digitized mouse pointer to correctly overlay the background image.

4.19 QRT

The QRT format is used by several exceedingly powerful ray tracing programs such as DKB Ray Trace and the QRT ray tracer. The format itself is quite simple, actually being a single file cousin to the SCULPT format.

QRT images are 24 bit-plane color only. A drawback of the QRT format is that image data written in this format is not compressed in any way. Therefore, QRT files can be quite large.

To read an image in the QRT format, select the QRT loader with the **Load Format** button, then select the **Load** button to cause the file requester to appear. Use the file requester to select a file in the QRT format.

4.20 SCREEN

The SCREEN loader is an example of the versatility of the ADPro loader/saver interface. It doesn't load an image from disk as the other standard loaders do.

Rather, it captures the rendered image data from another Amiga screen and converts this data for use by ADPro.

Selecting the **Load** button causes the SCREEN loader to make a list of the currently defined Amiga screens. Each screen will be listed by its Intuition screen name. If a screen does not have an Intuition screen name, then its name will be presented as **No Name**. If more than one screen has no Intuition name, each **No Name** will be differentiated by a sequence number.

Select a screen to capture by clicking on the button containing the screen's name. You may confirm that this is the screen you intend to capture by clicking on the **Show** button. The selected screen will be brought to front. A click of the left mouse button will bring ADPro back in front.

There are two ways to actually load the selected screen. First, you may select the **Grab** button after selecting one of the screen choices in the SCREEN loader control panel. The selected screen will be loaded into ADPro and can then be processed as if it were any other type of loaded image.

The first method of screen grabbing cannot allow you to grab a screen while that screen is showing menus. This is because you (the user) can't be in two places at one time (i.e., pushing the grab button and holding down the menu button in the screen you wish to grab).

The second method allows you to grab a screen including its menus.

When the SCREEN loader begins executing, it temporarily installs the hot key sequence, **Right Alt + Right Shift + Backspace**. When the SCREEN loader exits, it removes the hot key sequence from the system's input chain.

You can grab any screen including its menus by selecting the SCREEN loader and then bringing the desired screen to front, causing the desired menu to appear, and then depressing the hot key sequence. When you release the keys (while still holding down the right mouse button), the active screen will be loaded, including the active menus.

4.21 SCULPT

The SCULPT format was actually defined by Mimetics Corporation for use with their 24 bit-plane frame buffer. The format is better known as the one produced by Byte-By-Byte's SCULPT series of 3D modelling products.

The SCULPT format has no header information at all contained in the image itself. Therefore, in the SCULPT format, *you* must remember the exact dimensions of an image in order for it to correctly load.

After selecting the SCULPT loader and hitting the **Load** button, you will be

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

asked to enter a width and height of the image to be loaded. Remember, that the SCULPT format does not store the width and height as header information in the data files. Therefore, you must keep track of these values yourself.

The SCULPT loader can load a combination of three files to form a color image, or load a single file to create a grayscale image. After specifying a width and height, select which type of image (color or grayscale) you wish to load.

If loading a color image, you will be asked to specify three image files containing the red, green, and blue information for the image respectively.

ADPro will check the size of each of the three files you specified against the width and height you provided. If the size of one of the specified files does not match the size indicated by the entered width and height, the load will be aborted.

There is a naming convention for images stored in the SCULPT format. This convention dictates that each of the three raw image files comprising one SCULPT image share the same root file name and have the **red**, **grn**, and **blu** extensions for the red, green, and blue components, respectively.

When you specify the name of the red component, if you do not specify a filename extension or if the extension you specify is **red**, then the SCULPT loader will automatically generate the conventional extensions for the green and blue components.

The Mimetics frame buffer software also uses **ored**, **ogrn**, and **oblu** extensions to signify that the image in question is overscanned. The SCULPT loader will automatically generate these extensions where appropriate as well.

When loading a grayscale file, you will be asked to select only one file in the SCULPT format.

4.22 TEMP

The TEMP loader allows you to load the image currently in the ADPro temporary (undo) buffer into the primary image buffer. If no previous image was saved to the temporary buffer, then the image currently in ADPro will remain unchanged. For a discussion of saving to this temporary buffer, please read Section 5.23.

Besides using this buffer for "undo" operations, you can also use it for quick retrieval of the same image. One application of this technique would be to create various versions of the same image. You would save the image with the TEMP saver, make the first version's changes, and save it out with one of the savers. Press the **Load** button (which loads the original image from

the temporary buffer), make the second version's changes, and save those out. Continue pressing the **Load** button for subsequent versions.

Also, the TEMP loader retrieves the pixel aspect of the image in the temporary buffer.

To determine if image data is currently in the temporary buffer, press the **About** button on the main control screen. If an asterisk (*) appears to the right of the **Buffer Size:** value, then the temporary buffer is being used.

4.23 UNIVERSAL

The UNIVERSAL loader attempts to identify which file format a specific file is written in. If successful, the appropriate loader will be called automatically. The UNIVERSAL loader makes loading images of different file formats a bit more convenient since you do not have to manually switch between loaders. The penalty for this convenience is a minimal slow down due to bringing two loaders in from disk instead of one. But, with a reasonable hard disk drive, this slow down in inconsequential.

The UNIVERSAL loader attempts to find conclusive evidence identifying which file format a given file is written in. However, some file formats do not provide any way of locating conclusive proof of their identity. If the UNIVERSAL loader cannot locate conclusive proof, it does contain heuristics which enable it to guess (correctly) at a file type most of the time.

Some file formats, for example the SCULPT format, cannot be loaded from the UNIVERSAL loader at all.

The UNIVERSAL loader identifies file formats in a two step process. First, it uses file name extensions as a hint of which file formats to investigate first. Second, it checks the internal structure of the file against known facts about each file format. Table 4.1 lists some of the formats detectable by the UNIVERSAL loader including the file name extensions which it will recognize as hints.

If the UNIVERSAL loader cannot determine the file format for a given file, it will tell you. The UNIVERSAL loader will also inform you if it can determine which format a given file is written in, but you do not have a loader for that type. Some of these loaders are available as standard modules in the ADPro package, while others are from ADPro's companion package called the Professional Conversion Pack (PCP).

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Format	Extension
ANIM	.anim
DPIIe	.lbm
GIF	.gif
HAME	.hame
IFF	.brush
	.iff
	.ilbm
IMPULSE	.rgbn
	.rgb8
JPEG	.jpg
	.jpeg
	.jff
	.jfif
PCX	.pcx
QRT	.qrt
RENDITION	.6rn
TARGA	.tga
TIFF	.tiff
	.tif

Table 4.1: A list of some of the file formats detectable by the UNIVERSAL loader and the file name extensions recognized as hints.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 5

Savers

This chapter provides detailed descriptions of the use and operation of all savers which are included with ADPro. ARexx control of these savers is described in Chapter 7.

5.1 A2410

The A2410 saver gives you the ability to directly control the A2410 display board from Commodore. This board provides a high quality 8 bit-plane RGB video framebuffer.

NOTE You must run Commodore's TIGA initialization software before the A2410 can be used. See your A2410 documentation for more details.

The A2410 display board requires the presence of rendered image data containing from 1 to 8 bit-planes in a standard palette mapped configuration. That is, the rendered image data cannot be in one of the rendering modes unique to the Amiga such as HAM (Hold-And-Modify) or EHB (Extra-Halfbrite).

This saver is slightly different from some of the other savers compatible with ADPro in that it allows direct and continuous control over a hardware device, allowing multiple operations to be performed without exiting the saver. With a saver such as the one for GIF, the saver is invoked and a file gets saved. With the A2410 saver, the same image (or different parts of the same image) can be saved to the A2410 multiple times without leaving the saver.

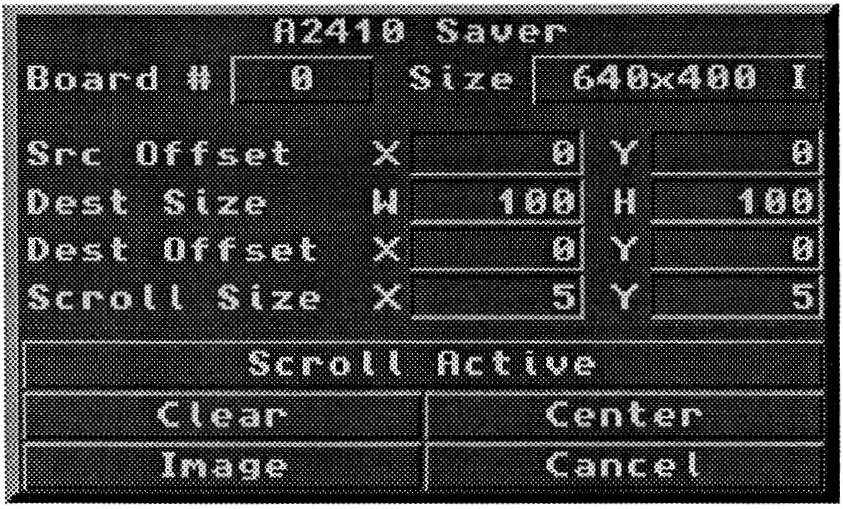


Figure 5.1: The A2410 saver control panel.

The A2410 saver control panel is shown in Figure 5.1. Along the top line, the button marked **Board #** allows you to toggle through which of possibly many A2410's you wish to control right now. If you have only a single A2410 in your system, this button will display a value of 0 only.

The A2410 contains two on-board crystals defining which two of several preset resolutions will be available. The button marked **Size** allows you to toggle between the two screen configurations that an A2410 can support at any one time. Table 5.1 lists the different standard screen resolutions possible with the A2410 along with the crystals required to implement those resolutions.

Screen Resolution	Interlace	Crystal Required (MHz)
640 x 400	Yes	14.318
736 x 480	No	26.6363
800 x 600	No	36.00
1024 x 768	Yes	44.98
1024 x 1024	No	67.88
1024 x 768	No	67.88
1024 x 1024	No	89.2108

Table 5.1: Resolutions supported by the A2410 and the crystals required to implement them.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The **Center** button automatically adjusts all of the sizing and offset controls of the A2410 saver so that the center of the image data will coincide with the center of the A2410, taking into account the currently selected screen resolution.

The button marked **Clear** will cause the entire screen to be cleared to black.

Selecting the button marked **Image** causes rendered image data to be transferred to the A2410. The amount of data transferred as well as where the image will be placed is controlled by the offset and size settings.

The **X** and **Y** values of **Src Offset** allow you to select the upper left hand corner in the source image data (the image in ADPro memory) from which to start transferring image data to the A2410. In general, when an image is smaller than the currently screen resolution, these values will be set to zero as they are in Figure 5.1. Setting these values to non-zero has the effect of "cropping" the image actually sent to the A2410 without actually disturbing the image data in ADPro memory.

The **Dest Size** values determine how large an area on the A2410 will be written into. This also determines how large an area from the source image (the image in ADPro memory) will be accessed. In the example of Figure 5.1, selecting the **Image** button will transfer a 100 by 100 pixel area from ADPro memory into a 100 by 100 pixel area in A2410 memory.

The **Dest Offset** values determine the upper left hand corner of the area which will be written in A2410 memory. This allows you to place an image into an arbitrary place in the A2410 display. For example, to center (by hand rather than using the **Center** button) a 320 by 200 pixel image on the A2410 display (when set to 736 by 480) you would set the **Dest Offset X** to 208 and the **Dest Offset Y** to 140.[1]

When scrolling is active, you can set the size of the basic scroll by placing values in the **Scroll Size** settings. These values determine how many pixels to the left or right (the **X** setting) or up and down (the **Y** setting) the image will move when the cursor keys are struck.

The **Scroll Active** button is necessary to tell the saver that you intend to use the cursor keys for scrolling when this control panel is displayed. Otherwise, the cursor keys will be used to move the cursor in the currently active input field. To use the cursor keys to scroll, select the button marked **Scroll Active** so that it remains highlighted. Now, depressing the cursor keys will cause the image on the A2410 to scroll in the specified direction. The area that will scroll is defined by **Dest Size**.

Note that the scrolling takes place only within the window you specified using the **Dest Size** and **Offset** fields.

[1]This is $(736 - 320)/2$ for the x offset and $(480 - 200)/2$ for the y offset.

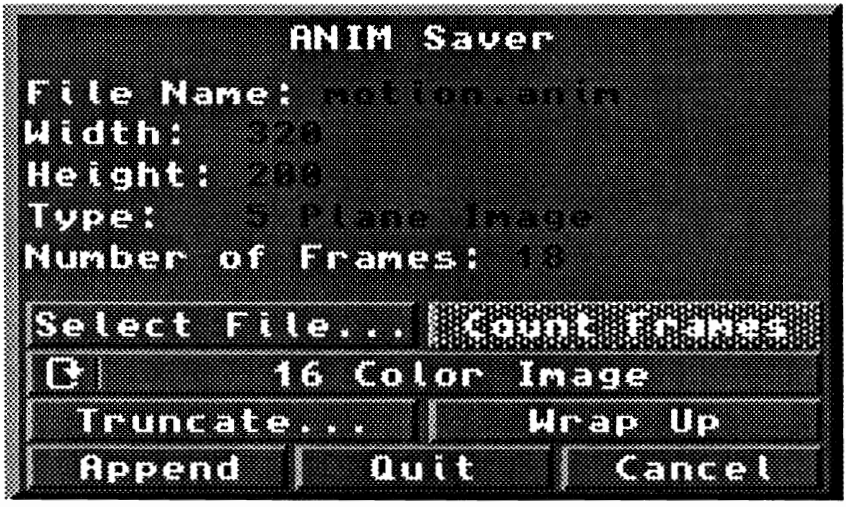


Figure 5.2: The ANIM Saver control panel. The **Count Frames** button has already been selected (**Number of Frames:** is not zero), causing it to be ghosted.

If you depress one of the cursor keys while the `Shift` key is also depressed, you will scroll three times the distance specified in the **Scroll Size** input fields. If you hold down the `Ctrl` key at the same time as one of the cursor keys, you will scroll to the far edge in the direction you specified.

5.2 ANIM

The ANIM saver allows you to save the current rendered image data (either the cropped screen or entire size) as the last frame of an IFF-ANIM file or as the first frame of a newly-created ANIM. Raw data (8 and 24 bit-plane IFF) is not currently supported in the IFF-ANIM file specifications.

After selecting the ANIM saver from the **Save Format** button and pressing the **Save** button, the ANIM saver control panel, as shown in Figure 5.2, will appear atop screen.

The first step you need to do is select the IFF-ANIM file onto which the current image data will be appended. Select the **Select File...** button. A file requester will appear to allow you to select the ANIM file. If you select a file that is a valid IFF-ANIM, the current rendered image data will be appended to it. If you enter the name of a non-existent file, the ANIM saver will create a new

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

IFF-ANIM file, using the rendered data as the first frame of the animation.

The upper half of the control panel is where information about the selected ANIM file will be displayed. To display the number of frames in the animation, select the **Count Frames** button. Note that this process may take some time to complete, especially for ANIM files that contain a lot of frames. Once you select an ANIM file, its filename will appear to the right of the **File Name:** label and the width, height, and type of each frame will be displayed next to the **Width:**, **Height:**, and **Type:** labels. After retrieving the frame count, this button will become unselectable (the button's image will be ghosted).

As previously stated, ANIMs can only be made up of rendered image data. The type of rendered data can be selected from the **Save Type** cycle gadget. Clicking on this button will toggle between **Cropped Screen Image** and *n* **Color Image**, where *n* is the number of colors in the rendered data.

If the current rendered image data does not match all of the selected ANIM's attributes (width, height, and type), a message will appear. The rendered data must be the same as what the ANIM expects, otherwise this saver will not be able to append it to the file.

Not only does the ANIM saver allow you to append an image to an ANIM file, it allows you to modify an existing ANIM file on disk.

The **Truncate...** button allows you to lessen the ANIM to fewer frames than it currently has. Selecting this button will bring up another panel with a integer input field. You can either enter a positive or negative number. If the number is positive, the ANIM will be truncated to that many frames. If you enter a negative number, that many frames will be removed.

The **Wrap Up** button will copy the first and second frames of the animation to the end of the file. This is part of the IFF-ANIM file format specification and is used to produce a smooth looping animation. This operation will close the ANIM file. This means that the next time you enter this saver, you will have to select a new ANIM file to work on.

To actually append the current rendered image data, select the **Append** button. The meter window will appear, showing the progress of the save. If any errors occur, a message will be displayed. If no errors occurred, the control panel will disappear.

A time-saving feature of the ANIM saver is that it remembers the position of the last frame in the ANIM file. This allows you to append successive frames without having to wait for the saver to find the end of a file for every new frame. For animations made up of a few frames, this feature might not be as evident as with a "multi-frame" animation.

To exit the saver without actually appending the image, select **Cancel**. The control panel will disappear, returning control back to the main screen. If you

want to use the time-saving feature just described, select this button to exit the control panel.

If you want to exit the control panel without saving the information about the last frame's position, select the **Quit** button. **Quit** will close the ANIM file, allowing you to use it in other programs.

Note that the ANIM saver will store the current **Screen Controls** settings with the saved frame.

NOTE If you want to play the animation with an ANIM viewer, you MUST select the **Quit** button after finishing all of your modifications with ADPro, select a different animation file using the **Select File...** button, select **Wrap Up**, or exit ADPro altogether. If you fail to do any of the steps, other programs will not be able to use the file.

5.3 BMP

The BMP format is the standard image file format used by Microsoft Windows. BMP files may come in one of four depths: 1, 4, 8 or 24 bit-planes. The BMP saver supports saving, raw, rendered, and screen image data. You will be asked to select one of the following:

- **Cropped Screen Image** — Selecting this mode will save the screen image data which would be displayed if you select the **ReDisplay** button. This choice is available only if you have rendered in a non-Amiga specific displayable mode.
- *n* **Color Image** — Selecting this mode will save the entire rendered image. The number of colors in the rendered image will be presented in place of the *n*. This choice is available whenever rendered image data is available in a non-Amiga specific format.
- *n* **Bit-Plane Raw Image** — Selecting this mode will save the entire raw image as either a 24 bit-plane color image or 8 bit-plane grayscale image.

Note that the BMP saver saves only in the uncompressed BMP format (although the BMP loader will read both compressed and uncompressed BMP files). This is because we have found that most Windows applications cannot read compressed BMP files. Also, saving 24 bit-plane BMP files using their compression will almost certainly produce a larger file than the original.

When the file requester appears, use it to select the name of the file under which a BMP file will be saved.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Note that you will benefit from use of the enhanced palette mode when saving rendered images in the BMP format.

5.4 CLIPBOARD

The CLIPBOARD saver allows ADPro to write pictures from the Amiga's Clipboard device. The Clipboard is a resource which can be shared between applications which support it. Note that text as well as images can be placed into the clipboard. The CLIPBOARD saver will write only picture type information.

Note that the Amiga's Clipboard device is not widely supported, and those applications that do support the Clipboard may not be expecting any other type of image data to be in the Clipboard except standard Amiga displayable images. Also note that the Clipboard device is often assigned to **RAM:**. This means that any image data written to the Clipboard will consume memory.

As an example of using the CLIPBOARD loader and saver, the icon editor (named IconEdit) supplied with Workbench 2.0 (and later) can copy from and paste to the Clipboard.

To load an icon into ADPro, drop the icon on the icon editor's window and select the **Copy** menu item. The icon is then written to the Clipboard. In ADPro, select the CLIPBOARD loader (and load the image). Instantly, the icon shows up in ADPro's image memory.

To save an icon from ADPro simply reverse the steps. Use the CLIPBOARD saver to save your prepared 4 color image (see Section 3.4.2). From the icon editor, select the **Paste** menu item. Instantly, the image shows up in the icon editor's image memory.

5.5 DPIIE

You must have rendered data available in order to save in the DPIIE format. To use the DPIIE format, the image must have been rendered with 256 colors. If you wish to save an image for use with Deluxe Paint II Enhanced with fewer colors than 256, you can use the standard IFF saver.

After selecting the **Save** button when DPIIE format is set, the file requester will appear. Use it to select a file name in which to save the rendered image.

Note that you can benefit from use of the enhanced palette mode when saving images for use with Deluxe Paint II Enhanced.

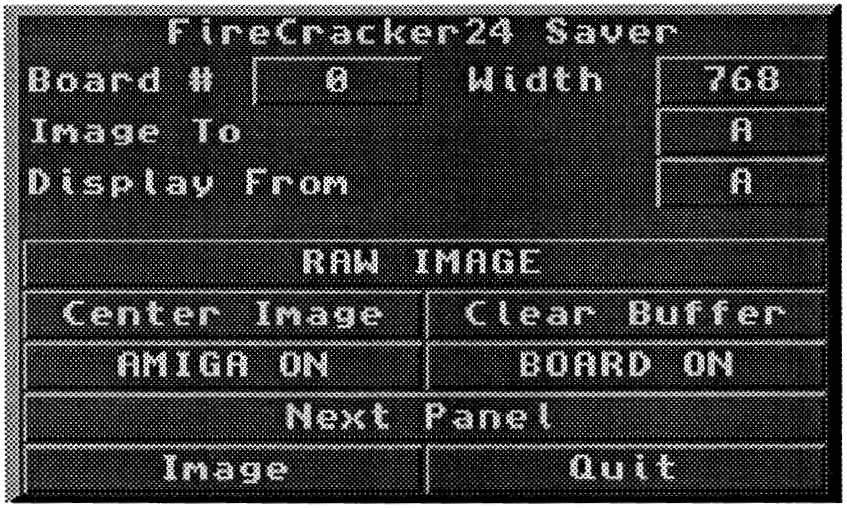


Figure 5.3: The first of two control panels for the FireCracker 24 saver.

5.6 FC24

The FC24 saver gives you the ability to directly control the FireCracker 24 display board from Impulse. This board provides a high quality 24 bit-plane RGB video framebuffer. Any work-in-progress within ADPro (both raw and rendered data) can be saved to the FireCracker 24.

This saver is slightly different from some of the other savers compatible with ADPro in that it allows direct and continuous control over a hardware device allowing multiple operations to be performed without exiting the saver. With a saver such as the one for GIF, the saver is invoked and a file gets saved. With the FC24 saver, the same image (or different parts of the same image) can be saved to the FireCracker 24 multiple times without leaving the saver.

The FC24 saver presents two control panels. The first is shown in Figure 5.3. Along the top line, the button marked **Board #** allows you to toggle through which of possibly many FireCracker 24's you wish to control right now. If you have only a single FireCracker 24 in your system, this button will display a value of 0 only.

The button marked **Width** allows you to toggle through the four board widths which the FireCracker 24 supports. These include 384, 512, 768, and 1024 pixels across.

The FireCracker 24 supports double buffering when set to either of its two lower resolutions (384 or 512 pixels wide). In its higher resolution modes (768

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

or 1024 pixels across) the FireCracker 24 does not support double buffering.

When double buffering is supported (i.e., the **Width** button set to 384 or 512), you can select which buffer will be written to by the next **Image** command by toggling the button marked **Image To**. This button toggles between an **A** and **B** setting corresponding to the two addressable buffers.

Similarly, you can choose which buffer is to be displayed by clicking on the button marked **Display From**. Like the button marked **Image To**, this button is enabled only when the FireCracker 24 width is either 384 or 512.

The button marked **RAW IMAGE** can be toggled between that setting and **RENDERED IMAGE**, provided that each type of image data is available. Using this button (and assuming both image data types are available) you can select which type of image data will be saved to the FireCracker 24.

The **Center Image** button automatically adjusts all of the sizing and offset controls of the FireCracker 24 saver so that the center of the image data will coincide with the center of the FireCracker 24, taking into account the currently selected width.

The button marked **Clear Buffer** will cause the entire buffer currently selected to be cleared to black.

The FireCracker 24 allows Amiga video to be looped through and combined with FireCracker 24 video for display on a single monitor. If you take advantage of this feature (to use a single monitor for both video outputs) then there are times at which you might want to disable either of the Amiga video or the FireCracker 24 video. The buttons marked **AMIGA ON** and **BOARD ON** can be toggled to enable or disable the respective video source.

Note that when the Amiga's video is disabled, all input (via mouse and keyboard, except for the cursor keys) will be disabled until either the left mouse button or the space bar is pressed. Note also that disabling the FireCracker 24 video while the Amiga video is also disabled will automatically reenable the Amiga video.

Selecting the button marked **Next Panel** takes you to the second FireCracker 24 saver control panel which is described below.

Selecting the button marked **Image** causes image data to be transferred to the FireCracker 24. The type of data is determined by the setting of the button marked **RAW IMAGE** in Figure 5.3. The amount of data transferred as well as where the image will be placed is controlled by settings found on the second FireCracker 24 control panel.

The second control panel for the FireCracker 24 saver is shown in Figure 5.4. It has several controls in common with the other FireCracker 24 control panel and such controls perform the same functions. They are repeated on both

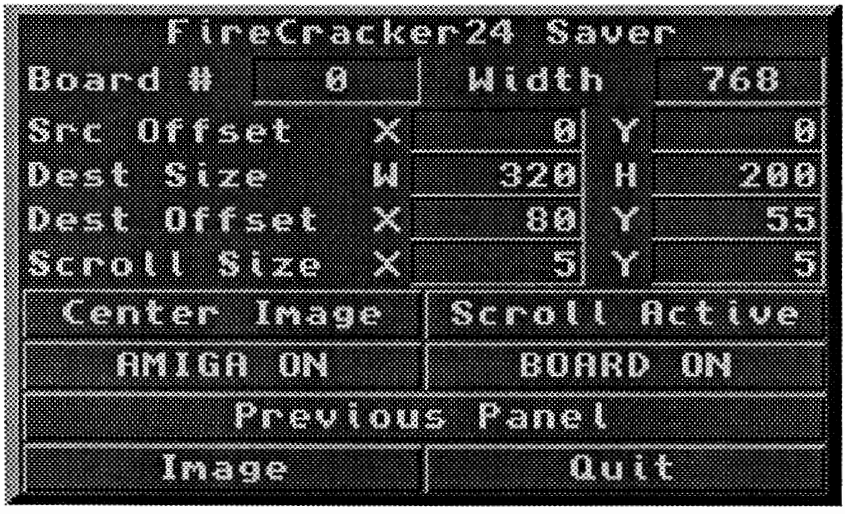


Figure 5.4: The second of two control panels for the FireCracker 24 saver.

panels for your convenience.

The **X** and **Y** values of **Src Offset** allow you to select the upper left hand corner in the source image data (the image in ADPro memory) from which to start transferring image data to the FireCracker 24. In general, when an image is smaller than the currently selected FireCracker 24 width, these values will be set to zero as they are in Figure 5.4. Setting these values to non-zero has the effect of "cropping" the image actually sent to the FireCracker 24 without actually disturbing the image data in ADPro memory.

The **Dest Size** values determine how large an area on the FireCracker 24 will be written into. This also determines how large an area from the source image (the image in ADPro memory) will be accessed. In the example of Figure 5.4, selecting the **Image** button will transfer a 320 by 200 pixel area from ADPro memory into a 320 by 200 pixel area in FireCracker 24 memory.

The **Dest Offset** values determine the upper left hand corner of the area which will be written in FireCracker 24 memory. This allows you to place an image into an arbitrary place in the FireCracker 24 display. For example, to center a 320 by 200 pixel image on the FireCracker 24 display (when the FireCracker 24 is set to 768 pixels wide) you would set the **Dest Offset X** to 224 and the **Dest Offset Y** to 141.[2]

When scrolling is active, you can set the size of the basic scroll by placing values in the **Scroll Size** settings. These values determine how many pixels

[2]This is $(768 - 320)/2$ for the x offset and $(482 - 200)/2$ for the y offset.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

to the left or right (the **X** setting) or up and down (the **Y** setting) the image will move when the cursor keys are struck.

The **Scroll Active** button is necessary to tell the saver that you intend to use the cursor keys for scrolling when this control panel is displayed. Otherwise, the cursor keys will be used to move the cursor in the currently active input field. To use the cursor keys to scroll, select the button marked **Scroll Active** so that it remains highlighted. Now, depressing the cursor keys will cause the image on the FireCracker 24 to scroll in the specified direction. Note that scrolling is always enabled when the previous control panel (the one shown in Figure 5.3) is displayed. This is because there are no input fields in that control panel, allowing the cursor keys to be used for scrolling purposes.

Note that the scrolling takes place only within the window you specified using the **Dest Size** and **Offset** fields.

If you depress one of the cursor keys while the `Shift` key is also depressed, you will scroll three times the distance specified in the **Scroll Size** input fields. If you hold down the `Ctrl` key at the same time as one of the cursor keys, you will scroll to the far edge in the direction you specified.

When Amiga video is disabled, scrolling is automatically enabled.

Selecting the button marked **Previous Panel** will return you to the panel shown in Figure 5.3.

5.7 FRAMEBUFFER

The FRAMEBUFFER saver requires the presence of raw image data. It writes this type of data to a Mimetics FrameBuffer, if you have one installed in your Amiga. The FRAMEBUFFER saver uses a device control library supplied by Mimetics Corporation to encode RGB data into the kind of data used by the FrameBuffer board. The encoding process (performed by the library supplied by Mimetics) requires approximately 700,000 bytes of free memory in order to function. Therefore, it may be necessary to use the **MAXMEM** tool type to ensure the availability of enough free memory to use this saver.

The saver will center the image on the FrameBuffer's screen if the image is smaller than the maximum size of the FrameBuffer's imaging area (768 by 484 pixels). If the image is larger than the FrameBuffer's imaging area, only the upper left portion of the image will be shown.

Any work-in-progress within ADPro can be previewed on the Mimetics FrameBuffer by selecting the FRAMEBUFFER saver and selecting the **Save** button.

5.8 GIF

The GIF saver requires the presence of rendered image data which cannot be in the Amiga specific formats of HAM (Hold-And-Modify) or EHB (Extra-Halfbrite).

Upon selecting the **Save** button you will be given the choice of which data to save. If the rendered data is in one of the allowable Amiga display modes, you can select **Cropped Screen Image** to save only the portion of the image which would be visible if you selected the **ReDisplay** button. Or, you can select the button describing the full size and number of colors of the rendered data to save the entire rendered image.

The file requester will then appear. Use it to select a file name in which to store the GIF image.

Note that you can benefit from use of the enhanced palette mode when saving images for use with applications which can read GIF.

5.9 HAME

The HAME saver gives you control over the HAME display device from Black Belt Systems. The HAME works in two modes:

1. In register mode, the HAME displays an image with 2 to 256 palette mapped colors. To produce a register mode image for use with the HAME saver, simply produce a rendering having between 2 and 256 colors and invoke the HAME saver. Use of the enhanced palette mode is suggested.
2. In HAME mode, the HAME displays an image using the same sort of tricks that the Amiga's own HAM mode employs to gain the impression of higher color content without greatly increasing the memory required to display such an image.

   HAME mode is supported in both ADPro's HAM mode as well as ADPro's HAM8 mode. When used in conjunction with renderings performed in ADPro's HAM8 mode, HAME mode can show colors from a 262,144 color palette. In either case, the use of the enhanced palette mode is suggested.

Creating a HAME mode rendering in ADPro's HAM mode is quite simple. Just perform the following steps:

1. Select the HAM rendering mode in ADPro and render an image as you would normally.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

2. Select the HAME saver and use it to either display or save an image in the HAME format.

Creating a HAME mode rendering in ADPro's HAM8 mode is slightly more complex. Follow these steps:

1. Enter the Palette control panel. Select HAM8 for **Colors (Total)**.
2. Select some number less than or equal to 60 for **Colors (Used)**. At 60 colors used, image quality will be maximized.
3. Ensure that **Offset Color Zero** is set to 0.
4. You are encouraged to utilize the enhanced palette mode.
5. Accept these settings on the Palette control panel.
6. On ADPro's main control screen, set the rendering type to CUST. Also select the desired screen settings, such as interlaced or overscan. Render the image by pressing the **Execute** button.
7. Select the HAME saver and use it to either display or save an image in the HAME format.

Upon entering, the HAME saver automatically detects if the image data is rendered in a palette mapped mode (2 to 256 colors) or in a HAM mode (HAM or HAM8). If in a palette mapped mode, the HAME saver automatically reformats the rendered image to create a register mode image. If the rendered image data is in a HAM mode, the HAME saver automatically reformats the rendered image data into a HAME mode image.

The HAME saver will inform you which type of image it has created via a text message contained in the saver's control panel. The control panel gives you the choice of displaying or saving the HAME image. If you elect to display the image, it will be displayed until you click the left mouse button. Note that unlike elsewhere in ADPro, you cannot scroll this display. If you elect to save the image, a file requester will appear for you to select the name under which the image should be saved.

The **Over** rocker switch on the HAME control panel allows you to set the type of overscan the image will displayed with. **Over W** will display only horizontal overscan. **Over H** will display only vertical overscan. Finally, **Over WH** will display both vertical and horizontal overscan. When the **Over** switch is set to simply **Over**, the image will be displayed with no overscan at all.

NOTE The HAME device is a *low resolution* device. That is, while the saver pays attention to the settings defining overscan, interlace, and choice of NTSC or PAL, it does not pay attention to the button choosing between high and low resolution.

5.10 HARLEQUIN

The HARLEQUIN saver gives you control over the display board of the same name from Amiga Centre Scotland. There are several Harlequin versions with differing capabilities. The HARLEQUIN saver is written to take advantage of the features of the top end version, including the ability to double buffer, genlock, and use an an alpha channel.

Figure 5.5 shows the control panel for the HARLEQUIN saver. If you have more than one Harlequin installed in your system, the button marked **Board #** allows you to select which board you will be controlling at the moment. The button marked **Width** allows you to cycle through the available display widths that the Harlequin supports. Note that if your board has a double buffering capability, changes to the width affect both buffers.

If your board supports double buffering, you can select which buffer will be written to by the next **Image** command with the button marked **Image To**. The button marked **A** to the right of the **Image To** button will become unghosted if you are currently working on grayscale data and your Harlequin has an alpha channel capability.

Assuming your board does, and you are currently working on a grayscale image, you can either render the image into the 24 bit-plane buffer or into the 8 bit-plane alpha channel. Selecting the button marked **A** causes the next **Image** command to write to the alpha channel rather than the main buffer.

The button marked **Display** allows you to select which 24 bit-plane buffer the Harlequin will display from if it is equipped for double buffering. The button marked **G** turns the genlocking capability of the Harlequin on or off for the currently displayed buffer. The button marked **A** turns the display of an alpha channel on or off for the currently displayed buffer, if your board is equipped with alpha channel capability.

To summarize, you can image to either buffer 0, buffer 1, the alpha channel for buffer 0, or the alpha channel for buffer 1. You can display from buffer 0 or 1, with or without genlock, and/or alpha channel.

The **Src Offset** input field represents where within your image the saver will begin fetching image data to send to the Harlequin. The upper left hand corner of your image is (0,0). The **Dest Size** defines the size of the rectangle (the "destination rectangle") of image data which will be sent to the Harlequin. In general, these values will be set to the size of the Harlequin's screen. However, you can set these values smaller to create a scrolling window on the Harlequin screen.

The **Dest Offset** defines where the destination rectangle will be placed on the Harlequin screen. In general this will be set to (0,0).

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

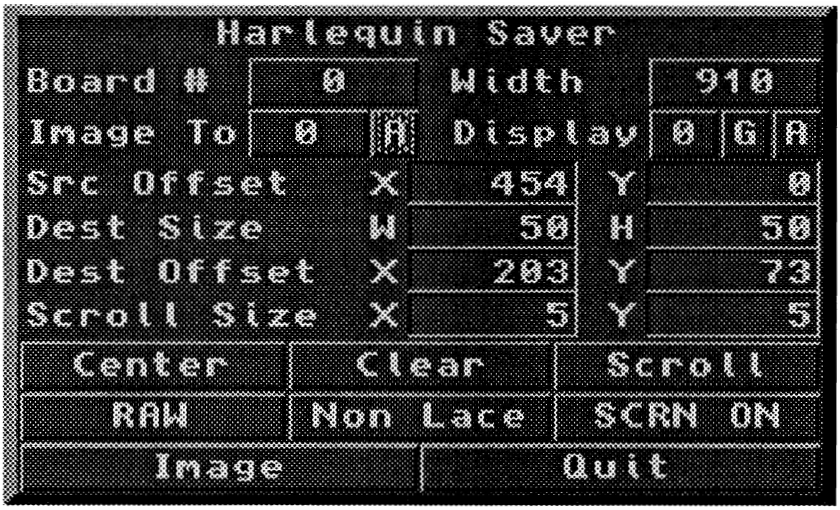


Figure 5.5: The ACS HARLEQUIN saver control panel.

The **Scroll Size** values determine how many rows or columns are shifted when scrolling is enabled. Scrolling is enabled when the **Scroll** button is selected. When enabled, the four arrow keys on your keyboard control scrolling. Depressing the left or right arrow key will scroll by the amount specified in the **Scroll Size X** gadget. Depressing the up or down keys will scroll by the amount specified by the **Scroll Size Y** gadget. Holding down the `Shift` key in conjunction with any of the arrow keys triples the corresponding scroll amount. Holding down the `Ctrl` key in conjunction with any of the arrows keys causes the image to scroll as far as possible in that direction. Again, scrolling is only enabled when the **Scroll** button is selected.

The button marked **Center** causes the offsets and sizes to be adjusted so that the image currently in ADPro's memory will appear centered on the Harlequin screen (at the currently selected width). The button marked **Clear** causes the screen selected with the **Image To** button to be completely cleared.

If raw and rendered image data is available, you can image either by selecting the appropriate setting on the button marked **RAW** in Figure 5.5. The rocker switch marked **Non Lace** toggles the Harlequin into and out of interlaced video mode. The **SCRN ON** rocker switch toggles on or off the Harlequin video output.

The **Quit** button causes the saver to exit and return control to the ADPro main screen. The button marked **Image** causes image data to be sent to the Harlequin subject to the values defined for offsets and sizes.

5.11 IFF

The IFF saver allows you to save the current raw, rendered, or screen image data in IFF-ILBM format. You will be asked to select one of the following:

- **Cropped Screen Image** — Selecting this mode will save the screen image data which would be displayed if you select the **ReDisplay** button. This choice is available only if you have rendered in an Amiga displayable mode.
- *n* **Color Image** — Selecting this mode will save the entire rendered image. The number of colors in the rendered image will be presented in place of the *n*. This choice is available whenever rendered image data is available even if the rendering mode is not directly Amiga displayable.
- *n* **Bit-Plane Raw Image** — Selecting this mode will save the entire raw image as either a 24 bit-plane color image or 8 bit-plane grayscale image.

Whenever a raw image is saved, ADPro includes Commodore standard **CLUT** chunks which represent your current color balancing settings. However, many programs which now support Commodore standard 24 bit-plane IFF files do not yet support **CLUT** chunks. To enable you to get exactly the image you intend, you can use the Apply_Map to permanently modify the raw data to include the effect of the **CLUT** chunks.

Raw images saved in the IFF format by ADPro also include an **ASDG** chunk which preserves the internal representation of your current color balancing settings. Reading and rewriting such a file in a non-ASDG program will more than likely cause the **ASDG** chunk to be removed. However, this means only that the color balance settings are lost and not that the data has been modified. You can reset the color balance settings by hand if you need them again.

After selecting which type of data to be saved by the IFF saver, the file requester will appear allowing you to specify the output file destination.

5.12 IMPULSE

The IMPULSE saver requires the presence of 24 bit-plane raw color image data.

You will be asked to select between RGBN and RGB8 formats. The RGBN format is a 12 bit-plane format in which ADPro will truncate the least significant four bits from each of the red, green, and blue components of the image. The RGB8 format is a 24 bit-plane format in which ADPro will preserve the entire raw image it has.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

After selecting which type of data to be saved by the IMPULSE saver, the file requester will appear allowing you to specify the output file destination.

While this format supports only raw image data, it does not contain its own color look-up tables. Therefore, in order to preserve the effect of any color balance settings you might wish to maintain, use the Apply_Map operator to transfer the effect of the color settings into the raw image data.

5.13 IV24

The IV24 saver gives you control over the GVP Impact Vision 24's display capabilities.

NOTE You must have an IV24 installed in your machine and you must have installed the `fye.library` in your `LIBS:` directory to make use of this saver. If these conditions are not met, selecting this saver will cause an error message to be displayed on ADPro's main screen.

Figure 5.6 shows the control panel which will appear when you execute the IV24 saver.

The **X** and **Y** values of **Src Offset** allow you to select the upper left hand corner in the source image data (the image in ADPro memory) from which to start transferring image data to the IV24. In general, when an image is smaller than the currently selected IV24 display size, these values will be set to zero. Setting these values to non-zero has the effect of "cropping" the image actually sent to the IV24 without actually disturbing the image data in ADPro's memory.

The **Dest Size** values determine how large an area on the IV24 will be written into. This also determines how large an area from the source image (the image in ADPro memory) will be accessed. In the example of Figure 5.6, selecting the **Image** button will transfer a 304 by 157 pixel area from ADPro memory.

The **Dest Offset** values determine the upper left hand corner of the area which will be written in IV24 memory. This allows you to place an image into an arbitrary place in the IV24 display.

Selecting the button marked **Image** causes image data to be transferred to the IV24. The size of the screen opened for the IV24 depends upon the settings of the four buttons located above the button marked **Center**.

You can open the IV24 with a high or low resolution screen, in interlace or not, in the NTSC or PAL video mode, or with overscan.

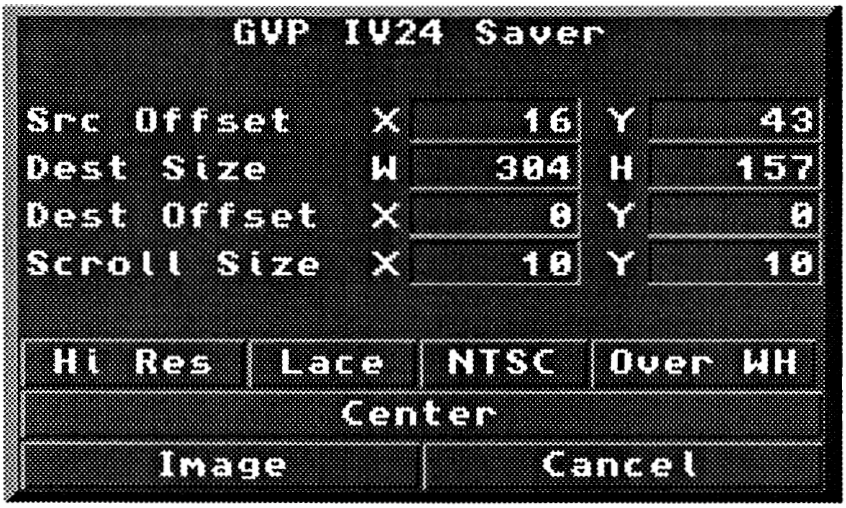


Figure 5.6: The IV24 saver control panel.

After pressing the **Image** button, you can use the cursor keys to scroll about the image. You can set the size of the basic scroll by placing values in the **Scroll Size** settings. These values determine how many pixels to the left or right (the **X** setting) or up and down (the **Y** setting) the image will move when the cursor keys are struck.

When viewing the image on the IV24, pressing the cursor keys will cause the image to scroll in the specified direction. Note that the scrolling takes place only within the window you specified using the **Dest Size** and **Offset** fields.

If you depress one of the cursor keys while the `Shift` key is also depressed, you will scroll three times the distance specified in the **Scroll Size** input fields. If you hold down the `Ctrl` key at the same time as one of the cursor keys, you will scroll to the far edge in the direction you specified.

The button marked **Center** will cause your image to be centered on the currently defined IV24 screen. It does this by setting appropriate values in the **Offset** and **Size** input fields.

5.14 JPEG

The JPEG saver can create files conforming to the JFIF file format standard. JPEG itself stands for the Joint Photographic Expert Group and is an image compression technology which dramatically reduces the size of an image for

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

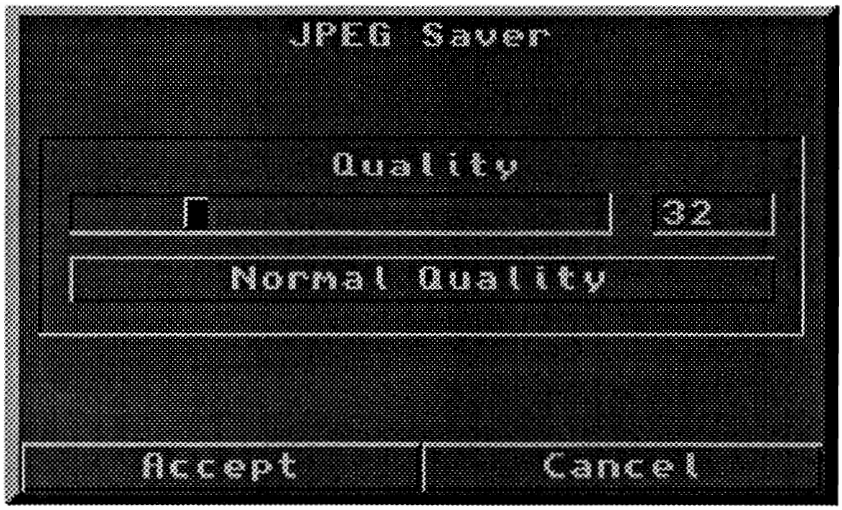


Figure 5.7: The JPEG saver control panel.

storage or transmission purposes. JFIF is a standard for constructing JPEG files.

Note that JPEG is a *lossy* compression method. That is, if you compress an image using the JPEG saver and then decompress it using the JPEG loader, the resulting image will not be exactly the same as the image you started with. The JPEG algorithm is designed with the human visual system in mind, so the changes it makes to an image are often unnoticeable. In fact, you can control the amount of change made to an image directly since you can control how much compression will take place. Higher compression means more change to the image.

The JPEG saver control panel, shown in Figure 5.7, allows you to control the amount of compression (which inversely affects image quality) and which type of JPEG encoding will be used. Note that the JPEG saver can save only raw image data.

The quality slider and input field ranges from 1 to 100. The higher this number is, the less compression will take place (which means that the quality of the resulting will be higher). A reasonable default value of 32 can often produce compression of 40 to 1 or better with little visually detectable loss.

The rocker switch marked **Normal Quality** in Figure 5.7 can be toggled to a **Boosted Quality** state. When in boosted quality mode, the quality slider again can range from 1 to 100, but there will be an order of magnitude less loss in the image. There will, of course, be a decrease in the amount of compression

as well.

The boosted quality mode is appropriate for when you demand as little loss as possible from the JPEG algorithm. In fact, at quality level of 100 boosted, no loss is contributed by the quantization step of the JPEG algorithm. Amazingly, even with such minimal loss, compression of 10 to 1 is still possible.

Selecting the **Accept** button will cause the file requester to appear. You can use the file requester to select the name of the JPEG file which will be written. JPEG files are often named with a suffix of "`.jpg`".

5.15 OPALVISION

NOTE The OPALVISION saver was developed by Opal Technology, and not ASDG, Incorporated. As such, you should contact Opal Technology for technical support. This module is included in the ADPro package under agreement with Opal Technology.

You must have version 1.6 or later of `opal.library` installed in your `LIBS:` directory. If you find that the saver crashes and you are running under Workbench 2.0 or later of the Amiga's operating system, type the following command at a Shell prompt to determine the version installed:

```
version LIBS:opal.library full
```

The OPALVISION saver gives you control over the OpalVision display board from within ADPro. The saver allows raw 24 bit color or grayscale images or rendered images from 2 to 256 colors to be displayed.

When the saver is activated, the OPALVISION saver control panel (as shown in Figure 5.8) is displayed. This control panel contains several buttons to control the image being displayed.

The uppermost rocker switch indicates type of image to be displayed. This switch allows you to toggle between the raw 24 bit color or grayscale image and the rendered image, if one exists. The rendered image data can contain 1 to 8 bit-planes (2 to 256 colors) in a standard palette mapped configuration. That is, the rendered image data cannot be in one of the rendering modes unique to the Amiga such as HAM, HAM8, or EHB.

Below this rocker switch are the resolution control rocker switches. These three switches control the horizontal resolution, vertical resolution and overscan, respectively. These buttons are much the same as the **Screen Controls** rocker switches on ADPro's main screen. The overscan button controls both the hoizontal and vertical.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

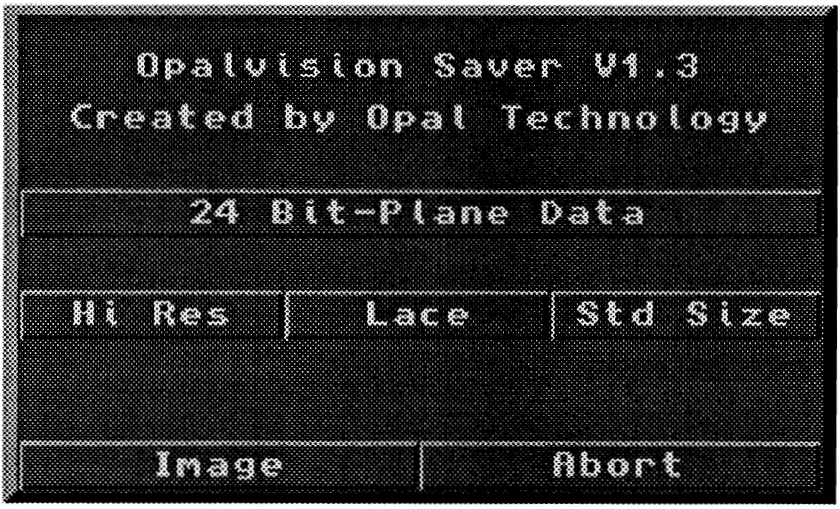


Figure 5.8: The OPALVISION saver control panel.

Key Sequence	Function
Del	Toggle cross hair pointer on/off
Arrow keys	Scroll display
Shift + Arrow keys	Scroll display in larger steps
Alt + Arrow keys	Jump to the extremeties of the image
n	Center screen under mouse pointer (if possible)

To view the image using the current settings, select the **Image** button. Once the image has been displayed, the saver will accept the following keyboard commands:

The arrow scroll keys are only applicable if the image is larger than the display screen size.

NOTE Unlike what is normally expected, the settings in the Balancing control panel will affect the rendered *as well as the raw data.* This means that you do not need to perform an Apply_Map operator to get these settings to affect the displayed raw image data.

To exit the saver, click the left mouse button and you will be returned to ADPro's main screen.

5.16 PCX

The PCX saver can save raw as well as rendered data in both color and grayscale. ZSoft Corporation, the developers of the PCX format, have extended the PCX specification to allow for 24 bit-plane color images, in addition to images ranging from 1 to 8 bit-planes. Note, however, that some applications may not currently handle the 24 bit-plane format of PCX.

The PCX saver cannot save an image rendered in one of the Amiga specific modes, such as HAM (Hold-And-Modify) or EHB (Extra-Halfbrite).

Upon selecting the **Save** button, you will be given a choice of which data to save. If the rendered data is in one of the allowable Amiga display modes, you can select **Cropped screen image** to save only the portion of the image which would be visible if you selected the **ReDisplay** button. Or, you can select the button describing the full size and number of colors of the rendered data to save the entire rendered image. Also, you may elect to save the raw image data.

The file requester will appear. Use it to select a file name in which to store the PCX image.

Note that you can benefit from use of the enhanced palette mode when saving rendered images for use with applications which can read PCX.

5.17 POSTSCRIPT

PostScript is a typesetting and page description language defined (and trademarked) by Adobe Systems Incorporated. ADPro can save a wide range of data in the form of a PostScript file. Such a file can be sent directly to a PostScript compatible printer or, in the case of Encapsulated PostScript, can be imported into other programs.

The POSTSCRIPT saver always requires the availability of raw image data. Upon selecting the **Save** button, ADPro will analyze the available raw image data to determine if it is an arbitrary color image, a grayscale image, or contains only shades of one specific primary color (such as cyan, yellow and magenta). This analysis will affect how the POSTSCRIPT saver will compose the final PostScript output.

The POSTSCRIPT saver itself is comprised of three control screens. The first of these screens is presented in Figure 5.9.

The format of the PostScript output can be either Encapsulated (EPS) or Non-Encapsulated PostScript. For direct printing purposes, select **Non-Encapsulated**. For inclusion of the saved PostScript data into other pro-

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

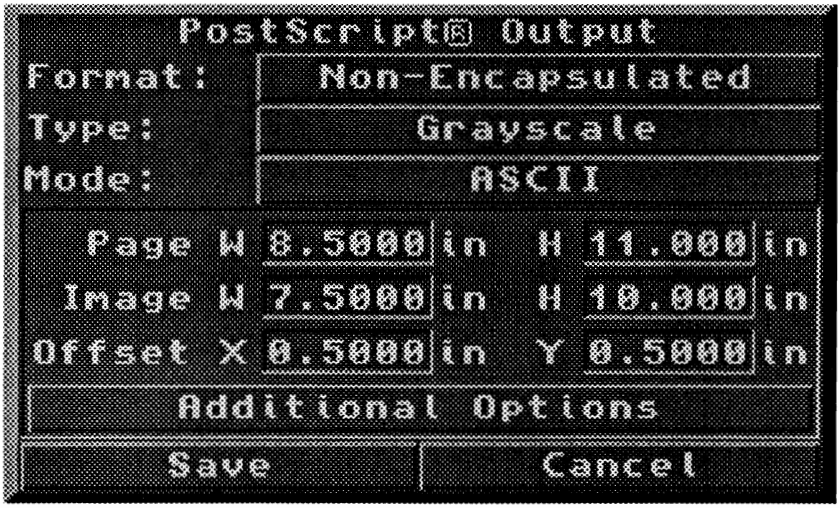


Figure 5.9: The primary POSTSCRIPT saver control panel.

grams, select **Encapsulated**.

The type of PostScript data to be saved can be grayscale, color, or separation data. If you select **Color** PostScript, the resulting file will make use of the PostScript color extensions and can be sent directly to a color PostScript compatible printer. Unless you have a printer capable of supporting color PostScript, the **Grayscale** choice is the appropriate selection. This will allow you to get a grayscale representation of your image right off of your PostScript printer. If you are processing color separation data (ADPro will determine this in the data analysis phase mentioned earlier), you can select a special setting which will include color separation specific instructions in the resulting PostScript file.

The mode rocker switch allows you to specify that the image data will be saved in either **Binary** or **ASCII** form. **Binary** form is much smaller than **ASCII** but can only be used with PostScript printers which are communicating via AppleTalk (registered trademark of Apple Computer) or some other similar printing-capable network. For most ADPro users, the correct setting is **ASCII**.

For non-Encapsulated PostScript output, the following options are available on this first control panel (all measurements are in inches):

- Page width and height defines the logical size of the PostScript page. Remember that optional features, such as registration and crop marks, exist and are printed outside the logical page. Therefore, you must leave room for them on the physical page should you want them.

- Image width and height defines the size of the box in which the image will be printed. The actual size of the image is effected by these values as well as whether or not the original aspect of the image will be preserved.
- The x and y offset of the image within the logical page specifies where the upper left hand corner of the image box will be relative to the upper left hand corner of the logical page.

The goal in designing ADPro's POSTSCRIPT saver was to provide the maximum amount of flexibility to the user. This comes at the cost of some confusion, however, since we provide so many ways of adjusting the output image. The possible confusion may be alleviated somewhat by Figure 5.10.

In Figure 5.10, notice that the registration and crop marks appear outside the logical page. Also note that the amount of overlap of the logical page and the crop marks is determined by the bleed setting. Be sure to set aside space on the physical page (by reducing the size of the logical page) if you wish to be able to see all requested crop and registration marks.

Pressing the button marked **Additional Options** takes you to the second POSTSCRIPT control panel shown in Figure 5.11.

Here, you may specify the angles and densities to be used in the PostScript output file.

The input field marked **C/R Dns** allows you to specify the Cyan density (all densities are in lines-per-inch) for color separation PostScript files or the Red density in color PostScript files. The input field marked **M/G Dns** allows you to specify the Magenta density for color separation PostScript files or the Green density in color PostScript files.

Similarly, the input field marked **Y/B Dns** allows you to specify the Yellow density for color separation PostScript files or the Blue density in color PostScript files. Finally, the input field marked **K/K Dns** allows you to specify the Black density for grayscale, color separation, and color PostScript files.

NOTE When printing grayscale image data (different from printing a color image in grayscale PostScript) only the black density is used.

The input field marked **C/R Ang** allows you to specify the Cyan angle (all angles are in degrees) for color separation PostScript files or the Red angle in color PostScript files. The input field marked **M/G Ang** allows you to specify the Magenta angle for color separation PostScript files or the Green angle in color PostScript files.

Similarly, the input field marked **Y/B Ang** allows you to specify the Yellow angle for color separation PostScript files or the Blue angle in color PostScript files. Finally, the input field marked **K/K Ang** allows you to specify the Black angle for grayscale, color separation and color PostScript files.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

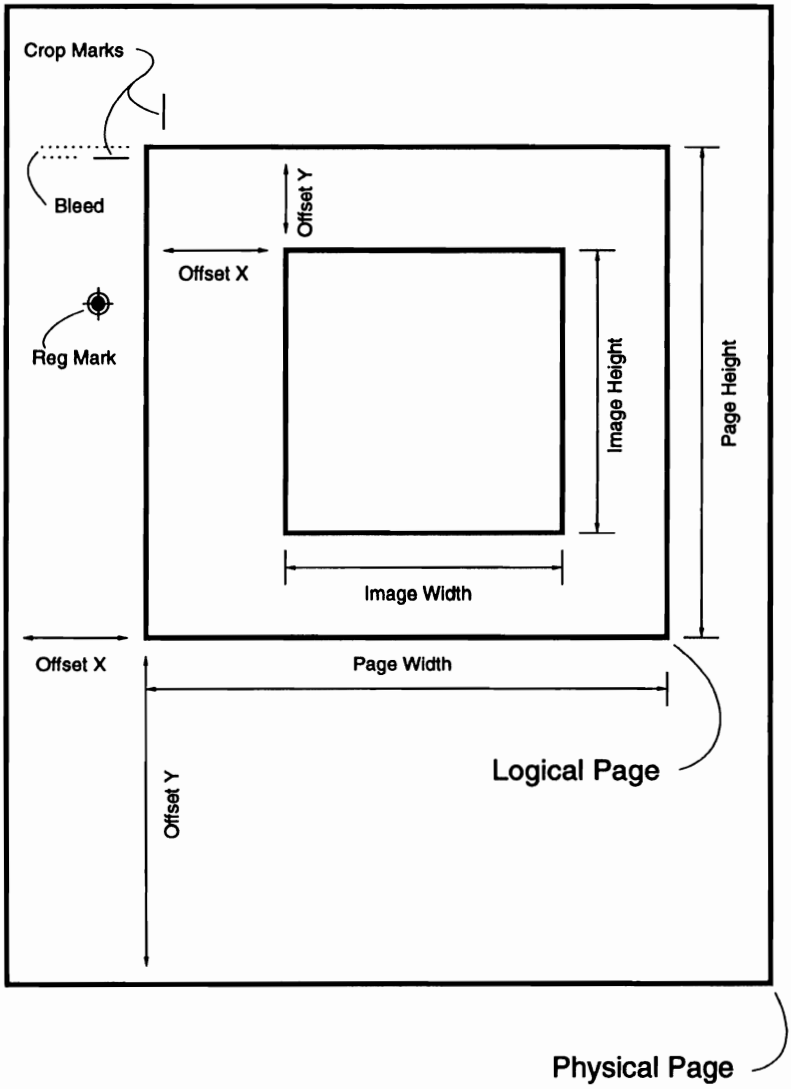


Figure 5.10: Various PostScript metrics.

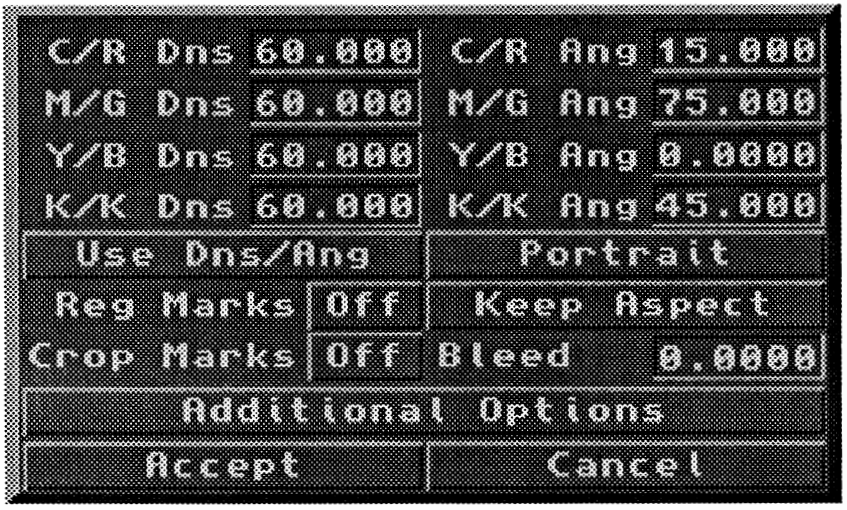


Figure 5.11: The second POSTSCRIPT saver control panel.

In Figure 5.11, the rocker switch marked **Use Dns/Ang** can be toggled between that setting and **Ignore Dns/Ang**. When set to **Use Dns/Ang**, the PostScript file will contain instructions telling the printer to use the angles and densities that you have specified. When set to **Ignore Dns/Ang**, the values you specify are ignored. Instead, the PostScript file contains instructions telling the printer to use its own internally stored default angles and densities.

You may wish to set this button to the **Ignore Dns/Ang** setting if you are working with a sophisticated printer which may have finely tuned settings already stored in its firmware.

The rocker switch marked **Portrait** can be toggled between that value and **Landscape**. Selecting **Landscape** rotates the logical page (see Figure 5.10) by 90 degrees.

Registration marks can be toggled on or off with the rocker switch labelled **Reg Marks**. Similarly, crop marks can be toggled on or off with the rocker switch labelled **Crop Marks**.

Note that crop and registration marks appear outside the logical page (see Figure 5.10). Remember to take this into account when specifying the page dimensions by ensuring that the logical page plus the space set aside for crop and registration marks can fit on the physical page.

The input field marked **Bleed** controls how much the crop marks (if present)

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

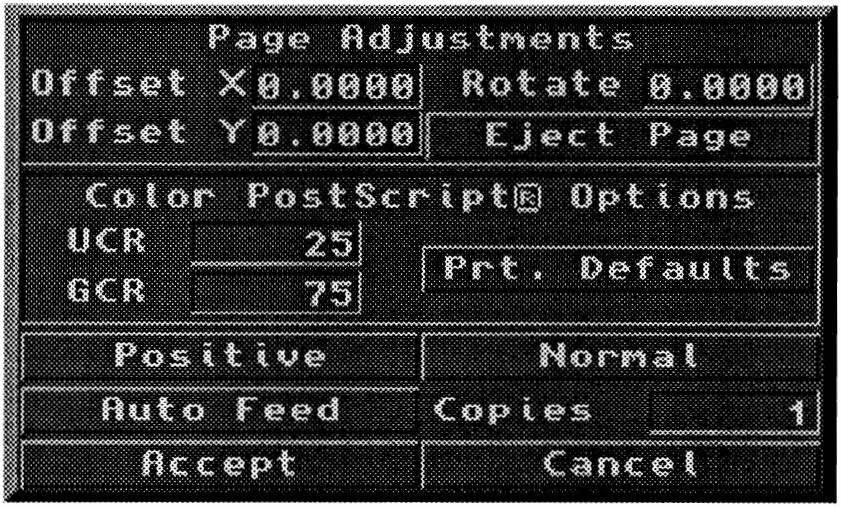


Figure 5.12: The third and final POSTSCRIPT saver control panel.

will overlap into the logical page.

The rocker switch marked **Ignore Aspect** in Figure 5.11 causes the image to be stretched as needed to fill the entire image width and height. Clicking on this button will toggle it to its other value, **Keep Aspect**. In this state, the image will be stretched to fill only one dimension of the specified image area. The other dimension will be stretched to match the scaling of the first dimension.

Pressing the button marked **Additional Options** leads you to the final POSTSCRIPT saver control panel (shown in Figure 5.12).

In this figure, the input fields marked **Offset X** and **Offset Y** allow you to specify the offset of the lower left hand corner of the logical page relative to the lower left hand corner of the physical page. The input field marked **Rotate** allows you to rotate the entire page about its lower left hand corner.

The rocker switch marked **Eject Page** in Figure 5.12 may be toggled between that state and **Don't Eject**. When set to **Don't Eject**, the PostScript file will contain instructions telling the printer not to eject the page when it is finished rendering inside the printer. This allows you to send additional pages which will be overlayed over the first. If set to **Eject Page**, the resulting PostScript file contains instructions telling the printer to eject the page as soon as it is completely rendered within the printer.

The rocker switch marked **Positive** allows you to specify that you wish a

positive or negative image rendered on the printer. To create a set of negative films (used for most printing applications) you would set this button to its other state, **Negative**. A special feature of the ADPro POSTSCRIPT saver is that it will actually produce a true negative even on PostScript printers which do not support the standard PostScript **negative** operator, such as the Apple LaserWriter series.

The rocker switch marked **Normal** in Figure 5.12 toggles between that state and **Mirror**. This button, used in conjunction with the **Positive/Negative** rocker switch will allow you to create mirror image negatives which is the standard used by professional printers. A special feature of the ADPro POSTSCRIPT saver is that it will actually produce a true mirror image even on PostScript printers which do not support the standard PostScript **mirror** operator, such as the Apple LaserWriter series.

Auto Feed specifies that the PostScript file should contain instructions telling the printer to take paper from its standard paper tray. Toggling this button to **Manual Feed** causes the PostScript file to contain instructions telling the printer to insist on a manually fed sheet of paper even if paper is available in its standard paper tray.

Finally, the input field marked **Copies** allows you to specify the number of copies that the printer will be told to make of each page output.

On each control panel, pressing **Cancel** will cause any changes made on that control panel to be ignored. You will be returned to the previous control panel. If **Cancel** is selected from the first control panel, then the POSTSCRIPT saver will exit and no image will be saved.

To accept changes made to a control panel, select the button marked **Accept**. This will cause the specified changes to be reflected in the next document you output. Pressing the **Accept** button from the first control panel will cause the file requester to appear.

Using the file requester you can select the name of the file into which the PostScript output will be saved.

To output the PostScript instructions to a PostScript printer connected to your Amiga computer through either the internal serial or parallel port, follow the steps below:

1. Blank out the **Drawer** input field.
2. Insert the name of the assigned device, such as **SER:** for serial or **PAR:** for parallel, in the **File** input field.

Remember to press the **Return** key after changing the contents of the input fields.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

5.18 PREFPRINTER

The PREFPRINTER saver allows ADPro to print the currently loaded image on any printer for which a Preferences printer driver (that supports strip printing) exists. If you have a color printer, you can print with 24 bit-planes of color accuracy. If you have a black-and-white-only printer, then you can print with 8 bit-planes of grayscale accuracy. If you were not using ADPro's printing capability, the Amiga's own color printing capacity would only be 12 bit-planes in color and 4 bit-planes in grayscale. Therefore, printing through ADPro allows you much better fidelity than if you had not printed through ADPro.

Also, the PREFPRINTER saver allows you to print poster-size images spread over multiple pages. In fact, ADPro itself does not impose any size limitation on the resulting print.

Selecting the PREFPRINTER saver causes the control panel shown in Figure 5.13 to appear. The control panel is dominated by two features. First, in the backdrop there is a ruled grid which represents the image to be printed, relative to page size and boundaries.

Second, in a movable window, are the many controls and displays that you use to get the printed results you want.

The ruled scale along the top and left edges of your screen automatically size themselves so that they will fill the width and height of the screen. The area depicted along these rulers is determined by several settings in the control window.

First, the overall dimensions of the depicted area are set using the two **Ruler Dimensions** gadgets. If the **Reference Scale** rocker switch is set to **English**, then the unit of measure used in the **Ruler Dimensions** is always inches. If the **Reference Scale** is set to **Metric**, then the unit of measure used in the **Ruler Dimensions** is centimeters.

Changing one of the **Ruler Dimensions** automatically changes the other dimension so that the resulting depiction will fit properly on one Amiga screen.

If you think of the ruled area as a canvas upon which you will place the image to be printed, the **Ruler Dimensions** define how big the canvas is. Making these values large means that you can depict many printer pages on screen at the same time. However, it means you have less visual resolution to make on-screen adjustments to the size and location of the area to be actually printed.

Therefore, if you are printing only a single page, it is best to set the **Ruler Dimensions** to reasonable values, such as 1.5 or 2 times the size of your printer's physical page size.

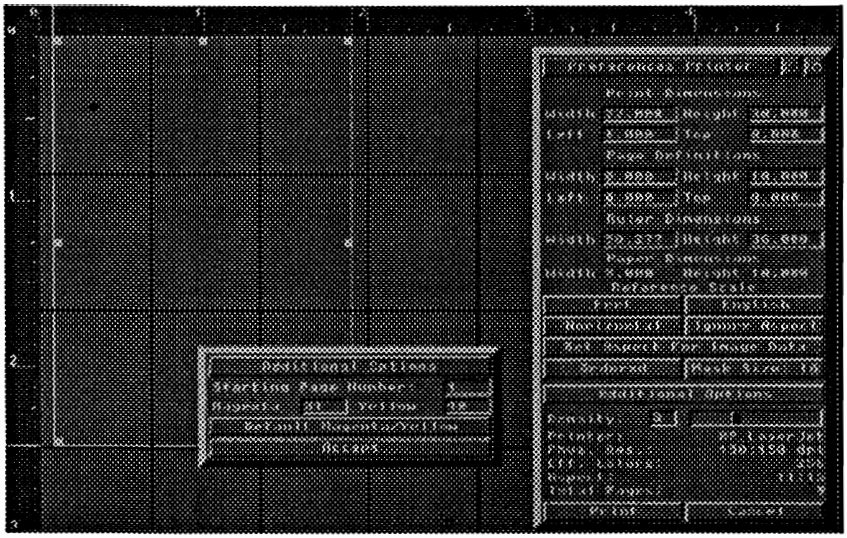


Figure 5.13: The PREFPRINTER control panel (also showing the Additional Options control panel). Note the print direction arrow in the middle of the first page to be printed.

As indicated above, you can set the unit of measure standard to either the English or Metric system. This can be done by clicking on the left hand button just under the words, **Reference Scale**.

Your choice of which printer is currently selected by Preferences will affect the operation of the PREFPRINTER saver in many ways. The PREFPRINTER saver shows you which printer it believes to be the currently selected printer in the text field marked **Printer:**. The PREFPRINTER saver asks Preferences how many dots-per-inch the currently selected printer is capable of and displays this information on the next line, labeled **Phys. Res.:** (for physical resolution).

Note that many Amiga Preferences printer drivers allow you to select several different printer densities. Selecting a different density from Preferences will be reflected by the PREFPRINTER saver. You can also set the printer's density directly from within the PREFPRINTER saver with the input field and slider marked **Density:**.

Also note that the PREFPRINTER saver interrogates Preferences to determine which printer to use (and other printer specific information) at the time the saver starts up. If you change any Preferences printer settings while the PREFPRINTER saver is running, these changes will not take affect until the next time you enter the saver.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Mask Size	Effective Colors	Effective Gray Shades
2	65	5
4	4097	17
8	262145	65
16	16777216	256

Table 5.2: Dither mask size versus effective color resolution.

Many dithering methods are available from PREFPRINTER. These are described below. In each case a larger dither mask (matrix) size produces a print which can represent a wider range of colors but will produce less spatial information per unit area of paper. Conversely, a smaller dither mask size can reproduce fewer colors (or shades) but more closely approximates the true resolution of your printer.

Another way of expressing this is the following: There is a tradeoff between printing many colors and printing in high resolution. Given a specific printer with a specific DPI capability, asking for many colors means using a larger dither mask size. A larger dither mask size cuts down on your effective resolution. (Table 5.2 shows the relationship between the size of the dither mask and the effective number of colors you will be able to reproduce.)

Note that this can work to your advantage when blowing a picture up in size. Blowing up a picture means that there are more dots to work with, which offsets the loss in resolution caused by a larger dither mask size. This, added to the benefits to be had by being able to reproduce more colors (or shades), means that your enlarged posters will look quite good.

Also note that many inexpensive printers, including most laser printers and dot matrix printers, have considerable dot gain. For example, a 300 DPI laser printer does not actually print dots which are $\frac{1}{300}$ of an inch in size. Rather, its dots will be much larger. This causes some dithers, such as the Floyd-Steinberg and Ordered dithers, to produce intensely over-saturated prints. Other dithers such as the two halftone dithers overcome this problem with low-end printers.

You might expect the values listed in Table 5.2 under the Effective Colors column to be 64, 4096, 262144, and 16777216, and under the Effective Gray Shades column to be 4, 16, 64, and 256. This might be correct for many other programs, but with PREFPRINTER the above is correct.

The best way to visualize the difference is with a Mask Size of 2. Assume you want to print a sample of all 256 possible shades of gray (where each shade has the same width as the others, and all are placed next to each other from darkest to lightest shade). With many other printing programs, this gray "ramp" would

not print as 5 equal-width shades of gray, but with PREFPRINTER it will. This will tell you that all 5 possible shades (for the example Mask Size of 2) are being used correctly.

To select a dither type, click on either the left or right side portion of the **Dither Type** button. Clicking on the left side will scroll backward through the available choices, while clicking on the right side will scroll forward.

The specific dithers available through PREFPRINTER include:

Floyd The Floyd (short for Floyd-Steinberg) dither is an error diffusion dither which can produce very good results on color printers with little dot gain and very good registration. If it produces a washed out print or particularly bad patterns, then try another dither.

Ordered The Ordered dither produces a regular repeating pattern which is often used for printing computer graphics. The Ordered dither is particularly susceptible to over-saturation due to dot gain on inexpensive printers.

Horizontal, Vertical, Fwd Diagonal, Bck Diagonal The Horizontal, Vertical, Fwd (Forward) Diagonal, and Bck (Backward) Diagonal dithers overcome many of the dot gain problems that the Floyd and Ordered dithers have with inexpensive printers. These dithers (especially the Diagonal dithers) are especially good for large posters.

Fwd Brick, Bck Brick The Fwd (Forward) Brick and Bck (Backward) Brick dithers are close cousins of the line-oriented dithers described immediately above. They produce an interesting effect.

Halftone A, Halftone B The Halftone dithers (Halftone A and Halftone B) differ in how they place a halftone matrix.

Halftone A causes the halftone matrix for each of the primary colors to be centered about the same point. This means that the primary colors will overlap completely, leaving a lot of white paper showing through. This may be appropriate for some better dye sublimation type printers or other color printers with good registration where the inks mix well.

Halftone B, on the other hand, staggers the halftone matrix of each primary color so that they do not overlap. This is similar in concept to traditional color offset printing. Halftone B may produce better results on printers whose inks do not mix well, and on printers with less than perfect registration.

The Halftone dithers can produce some extremely good results and compensate for the dot gain problems outlined above. Try both Halftone dithers to see which one is better for your particular printer.

ASCII The ASCII dither is a unique dithering method in that it does not require the printer to be graphics-capable (able to output graphics).

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Instead of using a pattern of dots, it uses actual ASCII characters. Although this method will not produce the most accurate images, it does produce acceptable results, especially when viewed from a distance.

Using the ASCII dither also allows owners of simple daisy-wheel printers to output their images to these types of printers, assuming a Preferences printer driver exists for it.

Note that the ASCII dither ignores the **Print Dimension' Left** value. This means that all of your ASCII printouts will start from the leftmost portion of each sheet of paper.

See Appendix C for samples of each of the above dither types.

Below the display of the effective number of colors can be found a display of the current print aspect. In Figure 5.13, the current aspect is given as 11:15. This means that the width of the area which will be printed is eleven fifteenths the size of the height of the area to be printed. This aspect is changed whenever you change the size of the select box on the backdrop, or change the width or height given under the **Print Dimensions** heading.

The aspect of the printed area can also be set correctly automatically by the PREFPRINTER saver by pressing the button marked **Set Aspect For Image Data**. This button causes the saver to interrogate the pixel aspect defined for both the image within ADPro's memory and the pixel aspect defined for the printer. It automatically computes some internal ratios so that a circle would print like a circle and a square would be square, regardless of the physical aspect of the printer's dots. *As long as ADPro's idea of the image's pixel aspect is correct*, hitting this button will allow you to print the image correctly, automatically.

Hitting the **Set Aspect For Image Data** button forces **Keep Aspect** mode (and changes the **Keep/Ignore Aspect** button to match). The **Keep/Ignore Aspect** button is the one marked **Ignore Aspect** in Figure 5.13. When this button is set to **Keep Aspect**, any change to the width or height of the select box will be matched with a corresponding change to the other dimension.

So, the quickest way to get a correctly proportioned print in the size of your choosing is to first click on **Set Aspect For Image Data** which automatically switches you into **Keep Aspect** mode, then drag out the select box to precisely the size print you want (or type in either the width or height you want and the PREFPRINTER saver will compute the appropriate value for the other dimension).

Remember, that the PREFPRINTER saver makes use of ADPro's currently defined internal values for pixel aspect and resolution. If these values are set inappropriately, the PREFPRINTER saver will be unable to correctly compensate for pixel versus printer aspect for you. For more information about

setting pixel aspect and image resolution, please refer to the Define_Pxl_Aspect operator..

The width listed under **Paper Dimensions** is set automatically when the PREFPRINTER saver interrogates Preferences. You must use Preferences to adjust this value. You can specify the physical paper height using the input field to the right. If you are using rolled paper, you can set the physical paper height to any reasonable value. If you are using sheet fed or folded paper, setting the paper height to larger than the physical height simply causes the print to extend over multiple pages. However, we suggest you enter the correct value for physical paper height so that the PREFPRINTER saver can accurately keep track of the number of pages being printed.

Now let's turn to the various controls which affect print size. Figure 5.14 shows the various page set-up dimensions that affect your print.

First, there are the margins physically required by your printer. You cannot modify these and they will be present on every page printed by the PREFPRINTER saver (including posters).

Subtracting any margins which might be required by your printer from the physical paper size yields the maximum printable area. These are presented to you by the width and height fields below the label **Paper Dimensions**. The PREFPRINTER saver does not allow you to change the paper width, although this can sometimes be done from Preferences. If you do change the paper width from Preferences, remember that you have to exit the PREFPRINTER saver and reenter it for those changes to be visible.

Unlike the physical page width, you can set the physical page height from within the PREFPRINTER saver. This was discussed previously.

Next come the margins you specify with **Page Definition Top** and **Page Definition Left**. You might want to use these fields to restrict the printable area to allow you to use narrow paper, for example, on a wide printer. In general, you can leave these fields set to 0 (and leave your **Print Definition Width** and **Height** set to the full size of the physically accessible page), which means that your print can occupy the total area physically possible.

Page Definition settings are applied to every page in a poster print.

Finally, the actual size of your print will be determined by the **Print Dimensions Width** and **Height**. The **Print Dimensions Left** will be applied to all of the left-most pages in a poster print. All of the top most pages of a poster print will be affected by the **Print Dimensions Top** setting.

The total number of pages which will be printed is given at the bottom of the control panel with the label, **Total Pages:**. Remember that PREFPRINTER's final idea of the page's dimensions, is determined by the settings under the **Page Definitions** heading.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

You can rotate your print either horizontally or vertically to make it better fit the physical size of your paper. This can often dramatically decrease the number of printed pages when printing posters.

Note that the setting specifying either horizontal or vertical printing will affect the order in which pages of a poster are printed. The first page to be printed will always be the one in the upper left hand corner of the bounding box of the poster. To assist you in determining the order the rest of the pages will be printed, a small arrow (indicating print direction) is printed in the center of the first page which will be printed.

You can set the first page to be printed from the Additional Options control panel. You can access this control panel by selecting the button marked **Additional Options**. Being able to set the first page printed is an essential feature for those doing poster prints since printers can often jam or run out of ink in the middle of a poster. To make effective use of this feature, count the number of good pages you have printed. The page you want to begin at is one more than the count of good pages.

Also available in the Additional Options control panel are ink compensation values which allow truer color printing. This allows blues, for example, to be printed as blue rather than as purple. The usage of the color correction gadgets are similar to those found in the color separation control panel. You can disable color correction by setting both the **MUCR** and **YUCR** to 0. A button is provided to restore the default values recommended by ASDG.

Selecting the **Print** button causes a print to begin. The **Cancel** button exits the PREFPRINTER saver without printing the image.

5.18.1 Setting the Preferences Paper Type

Using the PREFPRINTER saver, how you set the paper type in Preferences may be important.

If you have a single sheet **only** printer, such as a laser printer, skip this section.

If you have a printer which can take single sheets or continuous paper (such as fan-folded or rolled paper), PREFPRINTER can print multiple pages with either a gap or no gap between the pages. You can affect this by setting the paper type in Preferences to either "Fanfold" or "Single".

If you set "Single", PREFPRINTER will force a form feed between pages.

If you set "Fanfold", PREFPRINTER will not force a form feed, but some printer drivers may insert one themselves.

If you are printing a multi-column poster on continuous paper (fanfold or

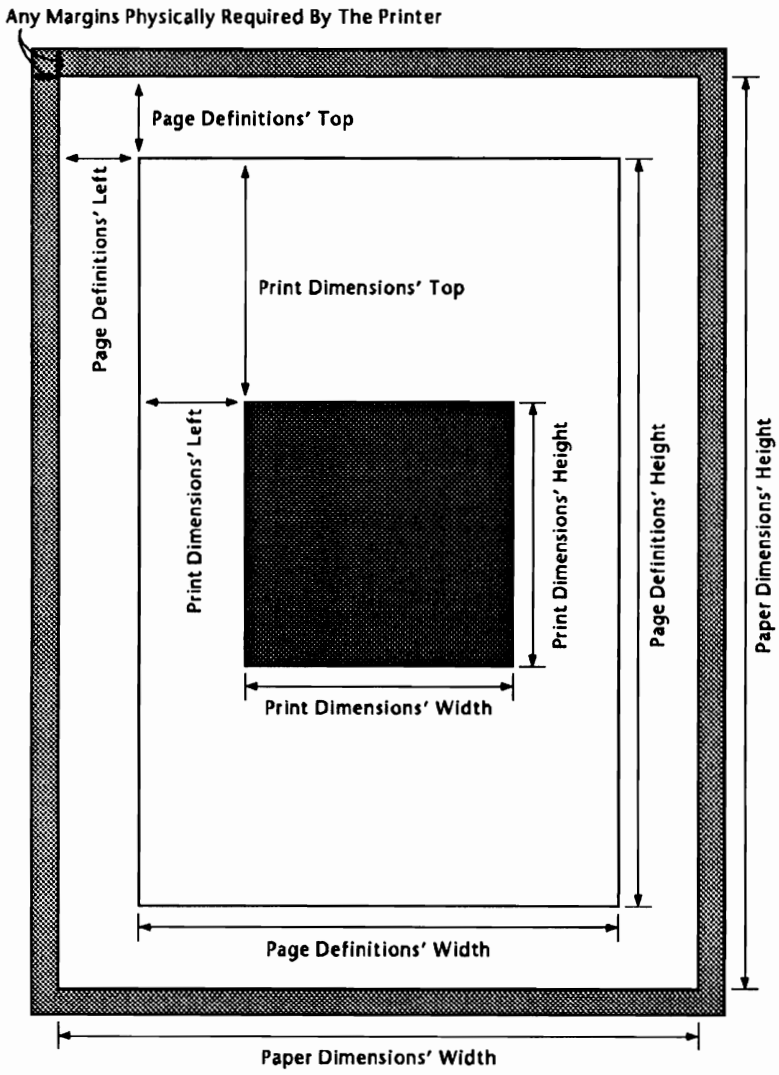


Figure 5.14: Various margins and measures associated with the PREF-PRINTER saver.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

roll), then you would set Preferences to "Fanfold" and define some top margin (**Print Dimensions' Left** or **Top** depending upon print orientation) so that the columns of your print will be separated. Also, when printing posters on continuous paper, you should set the **Page Definitions' Top** margin to 0 so that there is no gap between individual pages of a column.

5.19 QRT

The QRT saver requires the presence of 24 bit-plane raw image data. The file requester will appear. Use it to select the name of the file to be written in the QRT format. For more information about the QRT format, see Section 4.19.

5.20 RESOLVER

The RESOLVER saver gives you the ability to directly control the Resolver display board from DMI. This board provides a high quality 8 bit-plane RGB video framebuffer.

The Resolver display board requires the presence of rendered image data containing from 1 to 8 bit-planes in a standard palette mapped configuration. That is, the rendered image data cannot be in one of the rendering modes unique to the Amiga such as HAM (Hold-And-Modify) or EHB (Extra-Halfbrite).

This saver is slightly different from some of the other savers compatible with ADPro in that it allows direct and continuous control over a hardware device allowing multiple operations to be performed without exiting the saver. With a saver such as the one for GIF, the saver is invoked and a file gets saved. With the Resolver saver, the same image (or different parts of the same image) can be saved to the Resolver multiple times without leaving the saver.

The Resolver saver control panel is shown in Figure 5.15. Along the top line, the button marked **Board #** allows you to toggle through which of possibly many Resolver's you wish to control right now. If you have only a single Resolver in your system, this button will display a value of 0 only.

The resolution of the Resolver is determined by the configuration set up by the DMI configuration program. The Resolver board is capable of displaying non-interlaced images up to 2048 x 1024 pixels and interlaced images up to 2048 x 2048 pixels.

The **Center** button automatically adjusts all of the sizing and offset controls of the Resolver saver so that the center of the image data will coincide with the center of the Resolver, taking into account the currently selected screen

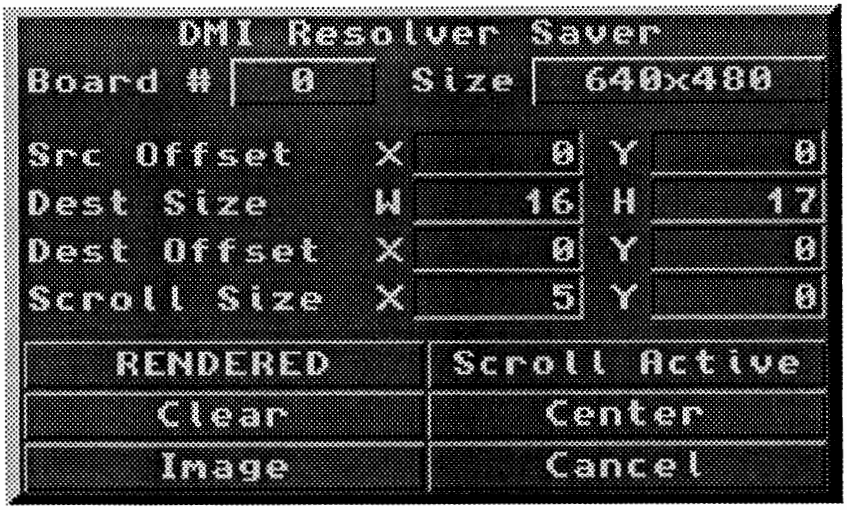


Figure 5.15: The RESOLVER saver control panel.

resolution.

The button marked **Clear** will cause the entire screen to be cleared to black.

Selecting the button marked **Image** causes rendered image data to be transferred to the Resolver. The amount of data transferred as well as where the image will be placed is controlled by offset and size settings.

The **X** and **Y** values of **Src Offset** allow you to select the upper left hand corner in the source image data (the image in ADPro memory) from which to start transferring image data to the Resolver. In general, when an image is smaller than the current screen resolution, these values will be set to zero as they are in Figure 5.15. Setting these values to non-zero has the effect of "cropping" the image actually sent to the Resolver without actually disturbing the image data in ADPro memory.

The **Dest Size** values determine how large an area on the Resolver will be written into. This also determines how large an area from the source image (the image in ADPro memory) will be accessed. In the example of Figure 5.15, selecting the **Image** button will transfer a 100 by 100 pixel area from ADPro memory into a 100 by 100 pixel area in Resolver memory.

The **Dest Offset** values determine the upper left hand corner of the area which will be written in Resolver memory. This allows you to place an image into an arbitrary place in the Resolver display. For example, to center (by hand rather than using the **Center** button) a 320 by 200 pixel image on the

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Resolver display (when set to 736 by 480) you would set the **Dest Offset X** to 208 and the **Dest Offset Y** to 140.[3]

When scrolling is active, you can set the size of the basic scroll by placing values in the **Scroll Size** settings. These values determine how many pixels to the left or right (the **X** setting) or up and down (the **Y** setting) the image will move when the cursor keys are struck.

The **Scroll Active** button is necessary to tell the saver that you intend to use the cursor keys for scrolling when this control panel is displayed. Otherwise, the cursor keys will be used to move the cursor in the currently active input field. To use the cursor keys to scroll, select the button marked **Scroll Active** so that it remains highlighted. Now, depressing the cursor keys will cause the image on the Resolver to scroll in the specified direction. The area that will scroll is defined by **Dest Size**.

Note that the scrolling takes place only within the window you specified using the **Dest Size** and **Offset** fields.

If you depress one of the cursor keys while the `Shift` key is also depressed, you will scroll three times the distance specified in the **Scroll Size** input fields. If you hold down the `Ctrl` key at the same time as one of the cursor keys, you will scroll to the far edge in the direction you specified.

5.21 SCULPT

The SCULPT saver requires the presence of 24 bit-plane raw image data. The file requester will appear three times. Use it to select the names of the red, green, and blue component files. Remember that you must explicitly keep track of the width and height of the image in order to make use of it again in the future.

5.22 SEPARATE

Executing the SEPARATE saver brings up ADPro's color separation control panel, as shown in Figure 5.16. Color separation is the process of turning red, green, and blue data appropriate for computer display into cyan, yellow, magenta and usually black data appropriate for print publication.

In order to color separate, raw color data must be available in memory.

ADPro supports 3 different types of color separations. These are:

[3]This is $(736 - 320)/2$ for the x offset and $(480 - 200)/2$ for the y offset.

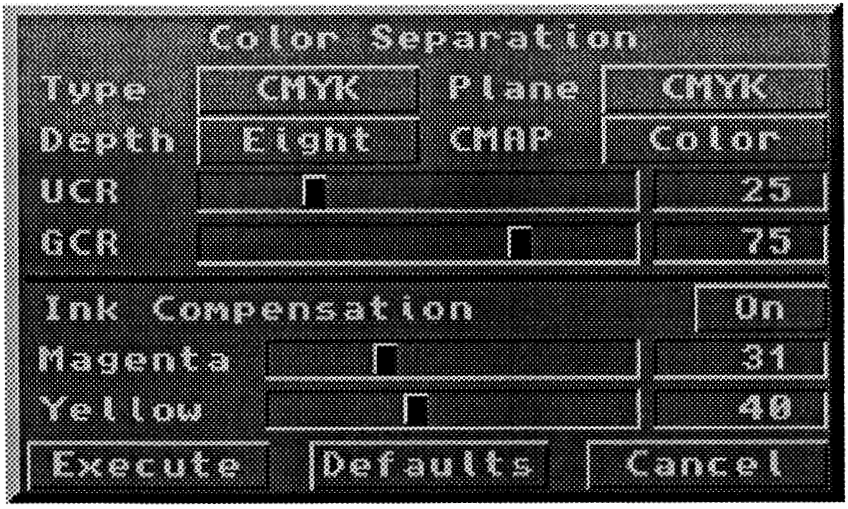


Figure 5.16: The SEPARATE saver control panel.

- RGB separations allow you to split the current color image into its component red, green, and blue planes. Each selected plane will be written to a separate file.
- Three color separations convert the current color image into cyan, yellow, and magenta planes without provision for a black plane. Three color separations are sometimes less expensive to bring to press but will not produce true gray tones which might be a part of the image.
- Four color separations convert the current color image into cyan, yellow, magenta, and black planes. Four color separations are what are generally used in the printing industry today.

Each of these different types of separation can be performed with an accuracy of 24 bit-planes or 12 bit-planes. A 24 bit-plane color separation can represent more than 16 million colors while a 12 bit-plane separation can represent only 4096 different colors.

The disadvantage of 24 bit-plane separations is that they can be quite large (however, 12 bit-plane separations can be quite large as well). The disadvantage of 12 bit-plane separations is that they can represent so few colors.

The choice of which to use (12 or 24 bit-planes) can depend upon many things. Such as:

- If your original image was an Amiga displayable format, the 24 bit-plane color data contains only 4096 colors (however, using ADPro's scaling

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

feature to reduce the size of the image will introduce additional colors to better blend the image). Therefore, you may not get any additional benefit from a 24 bit-plane separation.

- If your image contains many similar hues, such as the slowly changing colors in the human face in sunlight, then you may benefit from the dramatically greater dynamic range of a 24 bit separation. For example, a GIF image with 256 colors may contain a large number of shades of flesh tone. A 12 bit-plane separation can only represent at most 16 similar shades while a 24 bit separation can represent at most 256 similar shades.
- If you plan on using a standard Amiga printer driver and program to output the separations, chances are that the program you will use will not be able to deal with the files a 24 bit-plane separation would produce. However, if you plan on outputting your separation to a PostScript device, you may use either the 12 or 24 bit-plane separations.

Each plane of a 24 bit-plane separation contains 8 bit-planes. Each plane of a 12 bit-plane separation contains 4 bit-planes. You select between the two settings by clicking on the **Depth** rocker switch.

Using the **Type** rocker switch you can select from among the three types of separation that ADPro supports.

Within each type, you can select all possible planes or any one single plane by toggling the **Plane** rocker switch.

At your choice, the color separation files created by ADPro can contain a color or grayscale color map. This choice is determined by the **CMAP** button. When set to **Color**, each color separation file will have an appropriately colored color map included. For example, a cyan separation file will include a color map which ramps from black (where full cyan would be printed on paper) to full cyan (where white would appear on paper).

The **Color** setting is handy in that it allows you to get a visual impression of what a film would look like if printed singly in the appropriate color. And, in order to be compatible with ReSEP (another ASDG product which strips out 12 bit separations produced by older versions of Professional Page, replacing them with 24 bit separations), files must be separated with the **CMAP** rocker switch in the **Color** setting.

However, files separated with the **CMAP** switch set to **Color** are not technically "correct" separation files in that they contain full intensity cyan (for instance) where there should ultimately be white on paper.

By setting the **CMAP** button to **Gray**, a technically correct grayscale color map will be included with the separation file. When in this setting, white will appear in the separation file where white would appear on paper. Black will

appear in the separation file where full colored ink would appear on paper.

Note that separation files created while the **CMAP** switch is in the **Gray** setting will not be compatible with ReSEP. However, they will be more appropriate for use with the Inkmun Inks ((714) 948 - 2243) product for the Hewlett-Packard DeskJet 500.

When doing a four color (CYMK) separation, you can control the amount of **UCR** and **GCR** ADPro will perform. UCR (Under Color Removal) defines what percentage of color will be removed in order to compensate for the addition of the black plane. GCR (Gray Component Replacement) defines what percentage of the color removed will be added back in as black.

A four color separation with 0% UCR reverts back into a three color separation (i.e., there's no accommodation made for black). A four color separation made with 100% UCR will produce a color shifted image very similar in quality to early color photography.

In addition to **UCR** and **GCR** controls, ADPro also offers the user complete control over its **Ink Compensation** function. Printer's inks are not completely pure materials. For example, there is some amount of yellow mixed into the magenta ink. And, there is some amount of magenta which unavoidably is found in the cyan ink. The ink compensation values will correct for these impurities.

In general, leave the ink compensation values alone for best results. The ink compensation function can be completely disabled by turning it "off". You will notice that without ADPro's ink compensation function, a blue sky will print as purple. With the ink compensation function set at its default settings, blue skies are blue again.

To restore these default settings at any time, select the **Defaults** button.

When entering UCR/GCR or ink compensation values from the keyboard, the **Return** key toggles between **UCR** and **GCR** (when either of these two input fields is active) or between the **Yellow** and **Magenta** compensations (when either of these two input fields is active). **Alt + Return** can be used to toggle between the pairs.

5.22.1 Using ADPro Separations With Professional Page

Gold Disk's Professional Page (versions prior to 2.0) cannot manipulate images with a higher color resolution than the Amiga currently supports (12 bit-planes or 4096 colors). ASDG developed a program called ReSEP that allows 24 bit-plane separations created by Professional ScanLab, The Art Department, or ADPro to be merged with documents created in Professional Page.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Although the method is slightly circuitous, it works beautifully. Given below is the basic flow of operations:

1. In ADPro, perform any adjustments to the image you wish to make (for example, any color balancing or changes in contrast).
2. Render the image in an any Amiga displayable format and save this rendering to disk. In the steps that follow, we'll refer to this file as the "Amiga displayable" version.
3. Execute a color separation of the same image data and save this to disk. Note that the **CMAP** rocker switch must be set to **Color** for separation files to be compatible with ReSEP.
4. In Professional Page, construct the page as you wish placing text and image boxes where you wish them to go. Load the Amiga displayable version of the image (from the steps above) and place this image anywhere on the page as you would with any other Professional Page image box.

   You can scale, crop, or cover up this image in any way that you could with any other image box.
5. Execute a color separation from Professional Page, but save the results to disk.
6. In ReSEP, load the PostScript file generated in the previous step by Professional Page. ReSEP will identify all of the image boxes on the page. Using ReSEP select the ADPro separated files to replace the Amiga displayable version of the same image.

   When done, press the **Process** button. The output of ReSEP is a second PostScript file identical to the first, except that the selected images have had their 12 bit separations replaced with 24 bit separations.

5.23 TEMP

The TEMP saver allows you to easily and quickly make a "backup" copy of the image currently in ADPro. It uses the image buffer reserved for use by ADPro (the size of which can be displayed by clicking on the **About** button on the main control screen).

The TEMP saver, in conjunction with the TEMP loader, will allow you to always have a backup copy of your work in progress. Before you make a change which might produce unknown or undesirable results, first save the image using the TEMP saver. If after processing the image you are not satisfied with the results, simply use the TEMP loader to reload the unmodified image. If the image came out the way you intended, then proceed with more changes

or resave the new image with the TEMP saver and continue this step-wise process.

Although the TEMP saver can be used for "undo" operations, it can also be used as a way of setting up the use of the same image for manipulation without having to load the original image every single time. See the discussion of the TEMP loader for more details.

Also, the raw image's pixel aspect is also stored.

To determine if image data is currently in the temporary buffer, press the **About** button on the main control screen. If an asterisk (*) appears to the right of the **Buffer Size:** value, then the temporary buffer is being used. If you try to save to the temporary buffer and no asterisk appears, then there is not enough space in the buffer space (the number of bytes defined by **Buffer Size:**) to hold both the primary and temporary images.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 6

Operators

This chapter provides detailed descriptions of the use and operation of all operators which are included with ADPro. ARexx control of these operators is described in Chapter 7.

6.1 Apply_Map

Changes made with the **Color Controls**, such as **Brightness** or **Gamma**, do not actually change any of the raw image data. Rather, these changes affect only look-up tables through which the raw image data is filtered. Therefore, the affect of changes made with the **Color Controls** can be undone.

The Apply_Map operator applies the changes made with the **Color Controls** to the actual raw image data. It then resets the **Color Controls** to a neutral setting. The Apply_Map operator, once started, cannot be aborted.

Three uses of this operator come to mind:

1. It allows the range of the **Color Controls** to be extended, by allowing you to repetitively apply color control changes over and over again.
2. It allows for better results when merging images using the image compositing feature of ADPro. For instance, you might want to merge an image with bright color settings with an image with dark color settings. Using this operator on both images prior to compositing will allow you to get the desired results.
3. It allows other Amiga programs which can read 24 bit-plane IFF images but do not parse "color look-up table chunks" to get the image data you

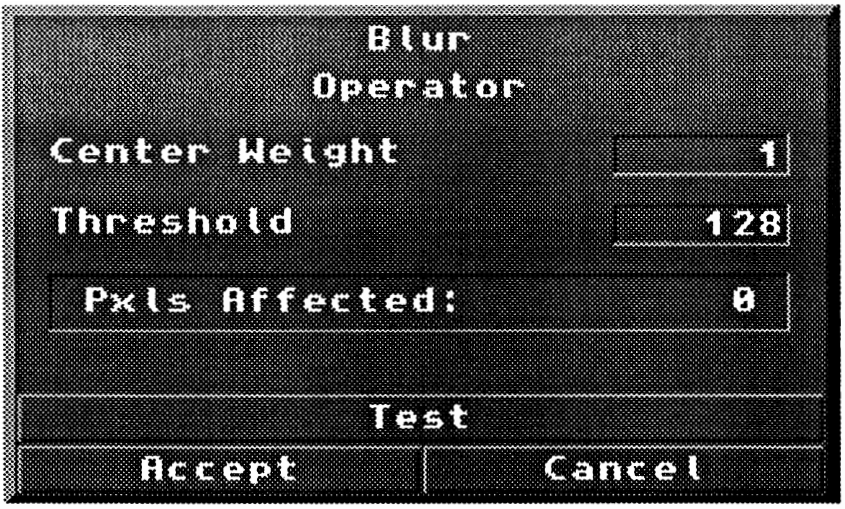


Figure 6.1: The Blur operator control panel.

intended.

6.2 Blur

Selecting the Blur operator causes the control panel shown in Figure 6.1 to be displayed. You can use this control panel to select or test the amount of blurring that will be done on an image.

The Blur operator allows you to blur certain pixels which differ from their neighbors by more than a user-definable threshold. Each pixel (weighted by a factor called the center weight) is averaged with the 8 surrounding pixels. If the resulting average differs from the actual pixel value by more than the user-defined threshold, the pixel is replaced by the average value. If the difference between the average and the original pixel does not exceed the threshold, the pixel is unchanged.

A center weighting factor from 0 to 16 may be specified. A lower center weight causes a more dramatic blur of the image.

A threshold of 0 to 255 may be specified. A value of zero forces all pixels to be blurred. A higher threshold will reduce the number of pixels which will be blurred.

The Blur operator allows a given set of values (for center weight and threshold)

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

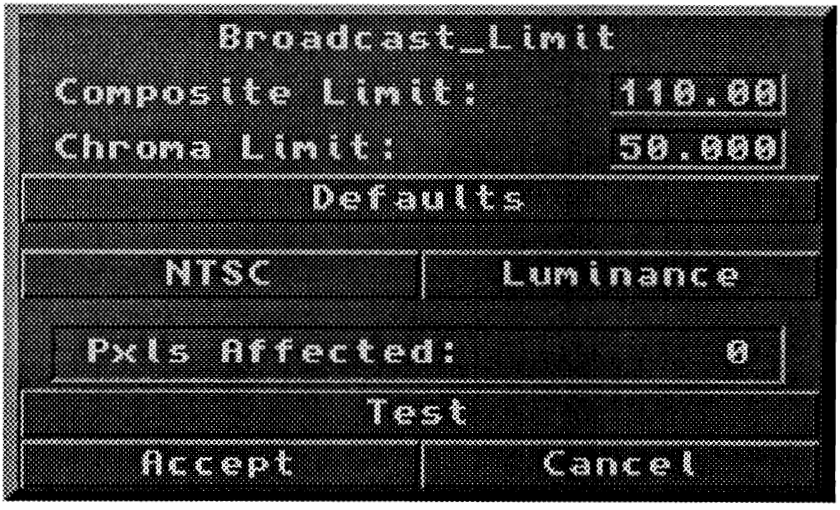


Figure 6.2: The Broadcast_Limit operator control panel.

to be "pencil checked" using the **Test** button. By selecting this button, all computations will be made, but no changes will be made to the original data. The number of pixels which *would have been* affected will be reported. You can use this feature to fine tune your selections of center weight and threshold.

Selecting the **Cancel** button will exit from the operator without modifying the data. Selecting the **Accept** button will apply the currently defined blurring operation to the raw data.

6.3 Broadcast_Limit

The Broadcast_Limit operator (shown in Figure 6.2) allows you to set the IRE limit of the composite video signal that would be generated if you were to display the currently loaded image on a NTSC or PAL display device. Video signal amplitudes are usually measured in IRE units, where 100 IRE is maximum white and 7.5 IRE is picture black (In PAL, picture black is the same as blanking black, which has an IRE value of 0).

When a picture reaches 120 IRE, video equipment may clip or distort the image being displayed. This may result in color errors in the area where the IRE limit exceeds 120. Colors, such as full-intensity yellow and cyan, encode to 131 IRE and, therefore, should be reduced before they are broadcast. An IRE value of 110 is a conservative limit for images which will be broadcast.

This will ensure the composite signal will not exceed 120 IRE even after being processed several times. Limiting the IRE to 100 is safer, but severely limits the range of colors in the image. A safe level for the chroma in an image is 50 IRE. If the chroma is higher, the image may be too saturated for an NTSC or PAL display.

The **Saturation/Luminance** rocker switch will allow you to switch methods for correcting out-of-bounds colors. Setting this button to **Saturation** will reduce the amount of color in offending pixel. The **Luminance** setting will reduce the brightness in each offending pixel.

A **Composite Limit:** value of 0 to 199.9 may be specified. A value of 0 will cause all pixels to become black, probably not what you had in mind. Specifying a value greater than 135 will have little or no effect on a picture (but will also not give any broadcast benefits).

A **Chroma Limit:** value of 0 to 99.9 may be specified. When using changes in luminance to correct for broadcast, a value of 0 will cause all pixels to become black. However, when using saturation to correct for broadcast, a value of 0 will cause the color image to be converted into a grayscale image. Specifying a value greater than 60 will have little or no effect on the image.

The **NTSC/PAL** rocker switch button allows you to switch between the default settings of the two major broadcast standards used today. The major difference between the **PAL** and **NTSC** settings are the value of picture black. In NTSC, picture black is 7.5 IRE, in PAL, picture black is 0.0 IRE.

The **Defaults** button resets the composite value to the 110.0 IRE standard, and the chroma value to the 50.0 IRE standard. The default values are the same for both NTSC and PAL machines.

The Broadcast_Limit operator allows a given set of values to be "pencil checked" using the **Test** button. By selecting the **Test** button, the computations will be performed but no changes will be made to the original data. The number of pixels which would have been affected will be reported in the **Pxls Affected:** text field.

Selecting the **Cancel** button will exit from the operator without modifying the data. Selecting the **Accept** button will apply the currently composite and chroma values to the raw data.

6.4 Collapse

The Collapse operator (shown in Figure 6.3) allows you to take a circular portion of your image (or the entire image itself) and "pull" it toward the center of the circle. This operator can be used to create caricatures of a

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

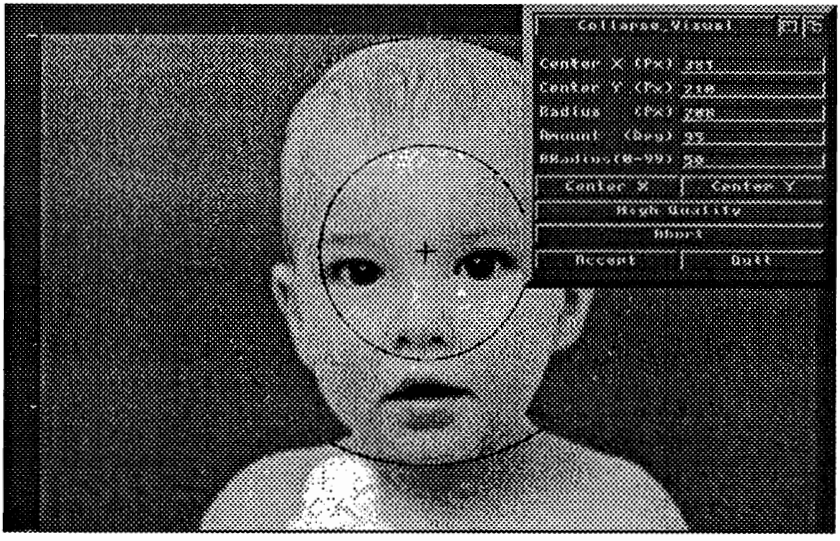


Figure 6.3: The Collapse operator control panel.

person's face or other artistic effect.

When the grayscale representation of your image is first being drawn on the screen, you can interrupt it by pressing the **Abort** button.

The center of the collapse circle (the area which will be collapsed) can be edited by dragging the circle's crosshair around on the screen or typing values into the **Center X (Px)** and **Center Y (Px)** input fields. You can center the circle horizontally, vertically, or both on the image by using the **Center X** and **Center Y** buttons.

The size of the collapse circle is described by its radius and defined in the **Radius (Px)** input field. You can also click virtually anywhere on the screen to adjust the circle's size. The circle will expand to contract to the location of the mouse pointer.

The amount of collapsing can be controlled by the value in the **Amount (Deg)** input field. The higher the amount the greater the effect, and vice versa.

The area near the outside edge of the circle can be "blurred" by specifying a value in the **BRadius(0-99)** input field. This value represents a percentage of the radius (starting from the outermost part of the circle going inward) to be blurred. A value of 0 means no blurring, while a value of 99 means that the entire circle will be blurred.

The quality of the effect can be controlled by toggling the quality rocker switch

between **Fast** and **High Quality**. **Fast** works faster, but may not produce as precise an image as **High Quality**.

Press the **Accept** button to perform the collapse operation, or **Quit** to return to the main screen without performing the operation.

6.5 Color_To_Gray

The Color_To_Gray operator provides three very important benefits beyond simply turning color images into grayscale images:

1. The method used within ADPro to convert from color into grayscale can actually produce grayscale data of considerably better quality than the original color data. For example, the Sharp JX-100 portable scanner is accurate in grayscale to 6 bit-planes and 18 bit-planes in color. However, when ADPro converts an 18 bit-plane color image into grayscale, it produces a grayscale image which is accurate to 8 bit-planes.
2. ADPro renders extremely well in grayscale due to its dithering capabilities and the fact that 16 grayscales cover the range of gray proportionately better than 16 colors cover the spectrum of all possible colors. For this reason, a high resolution interlaced gray image rendered with ADPro can look very nearly photographic, even given the Amiga's limited display capability.
3. This feature provides a way to convert color images into high quality grayscale images for the purpose of black and white desktop publishing. While color DTP is just now coming into its own, DTP is still predominantly grayscale.

Selecting the Color_To_Gray operator causes the control panel shown in Figure 6.4 to be displayed. You can use this control panel to select the weights with which the red, green, and blue components of the image will be factored into the color to grayscale conversion.

Two pre-programmed weighting schemes are provided. These can be enabled by selecting either the **Average Weights** or **Luminance Weights** button. By selecting the button marked **Average Weights**, each of the red, green, and blue components of the image will be given equal weight as the image is converted to grayscale. In essence, each gray pixel will be the average of the red, green, and blue color pixel components.

Selecting the button marked **Luminance Weights** will set up the weighting scheme used by the NTSC television standard to convert a color broadcast signal for display on a black-and-white television receiver.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

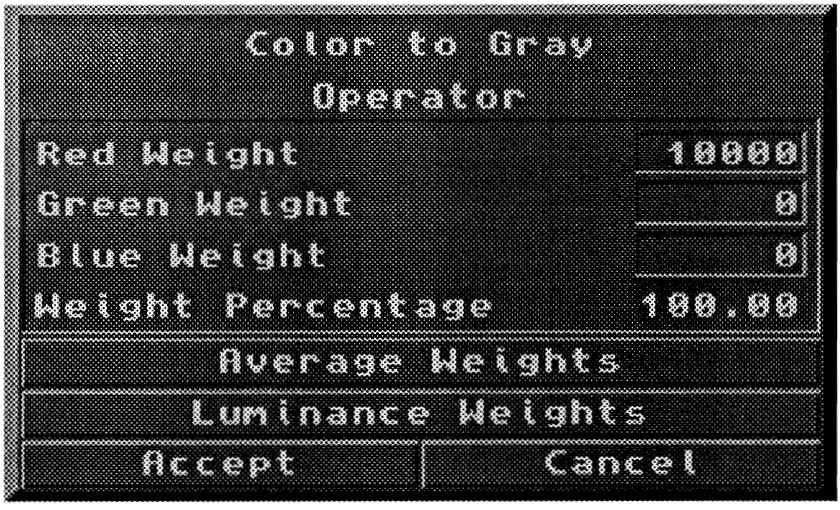


Figure 6.4: The Color_To_Gray operator control panel.

If you wish to specify your own weights, you may do so by entering values into the three integer input fields supplied. The values you enter represent the fractional contribution that each color will have in the final averaging process. These values are scaled such that 0 represents 0 percent weight while 10000 represents a 100 percent weight. The total weighting is given below the three integer input fields and may exceed 100 percent.

In Figure 6.4, a weight of 10000 has been given to the red component while a weighting of zero has been given to the other colors. This will cause pixels with a large red component in the original will be shaded lightly in the resulting gray scale image. Pixels with very little red will be shaded darkly in the resulting gray scale image.

A total weighting percentage may exceed 100 percent. Pixels in the resulting grayscale image are clipped to the range of 0 to 255. Negative weights may also be specified.

Selecting the button marked **Accept** causes the currently defined conversion to take place. Selecting the button marked **Cancel** exits the operator without changing the current image in memory.

6.6 Colorize

This operator is an alternative to Color_To_Gray in that both operators will reformat grayscale image data into color image data within ADPro's memory. However, Color_To_Gray simply creates grayscale data in the format of color data, whereas Colorize will actually color in the grayscale image.

Recall that grayscale image data is stored in 256 shades within ADPro.

Using the **Random** setting, the 256 shades of gray will be converted into 256 colors chosen at random. The color choosing is not completely random as this would produce an unrecognizable image. Rather, the random color choices will center around the color you specify in the requester (in the **Base Color** input fields).

You can convert grayscale into color using the currently loaded color palette if you select the **Currently Loaded** setting. You can load a palette in from disk prior to executing this operator if you wish to match a specific palette. Remember that in a grayscale image, BLACK is generally color register 0 and that WHITE is generally color register 255.

You can produce a false colorization with the **False Color** setting. A false colorization puts cold blues in the darker range, and hot reds in the brighter areas.

You can produce a wide range of effects with the **HSV Transitional** setting. These will be explained later.

You can specify a base color using the integer input fields provided. The **Base Color** is used by the **HSV Transitional** and **Random** settings.

NOTE The base color is used to set parameters for the **HSV Transitional** colorization, which means that setting the **Base Color** to specific values like BLACK will cause an entirely BLACK colorization (probably NOT what you had in mind).

When **Vary Hue** is selected, the value and saturation of the colors used in the colorization will be taken from the **Base Color**. The spread of hues created can range between 0 and 10000. Note that hues range from 0 to 360. Specifying a value larger than 360 will result in a cyclic repetition of colors.

When **Vary Saturation** is selected, the hue and value of the colors used in the colorization will be taken from the **Base Color**. The spread of saturation which will be used can range from 0 to 1.

When **Vary Value** is selected, the hue and saturation of the colors used in the colorization will be taken from the base color. The spread of value which will be used can range from 0 to 1.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

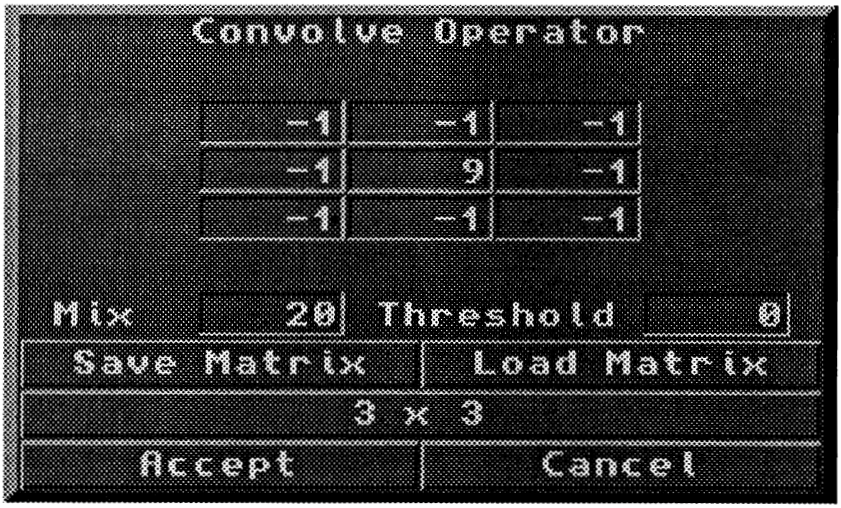


Figure 6.5: The Convolve operator control panel.

With all types of colorization, you can select a range of gray shades which will be affected. The range always flows from the number on the left, upwards to the number on the right. This means that you can specify a range which wraps around the 255 (white) mark back to the 0 (black) mark. Shades of gray which are outside the range can either be left alone (a **Fill** value of -1) or set to the specific intensity of gray (0 to 255), which is typed into the **Fill** input field.

For example, you might specify a false colorization between shades 0 and 127 with the fill value set to -1. This means any pixel which had a gray shade between 0 and 127 (inclusive) will be false colored in the new color image. Pixels from 128 to 255 will be the same shade of gray they were originally.

If you wished to colorize gray shades starting at 250 and wrapping around to 20, then you would specify shades 250 and 20 in the **From** and **To** input fields, respectively.

6.7 Convolve

The Convolve operator is a general purpose convolution filter which can apply 3x3 or 5x5 convolution matrices to the image data. Convolutions can perform many different image processing functions by varying the values comprising the convolution matrix. The control panel for this operator is shown in Figure 6.5.

As configured in Figure 6.5, the Convolve operator is ready to perform a sharp-

ening function. Determining the values to use in the convolution matrix requires either much experimentation or a knowledge of image processing theory. To save you from either of these, we provide a large number of well-known convolution matrices with the ADPro distribution.

To load a convolution matrix, use the button marked **Load Matrix**. Selecting it will cause the file requester to appear. Use the file requester to select the convolution of your choice. Similarly, a given convolution matrix can be saved using the button marked **Save Matrix**.

The rocker switch marked **3 x 3** in Figure 6.5 is used to switch between 3 x 3 convolution matrices and 5 x 5 matrices.

The **Mix** and **Threshold** values interact as follows: After a given pixel has been convolved, the new value for that pixel is compared with the old value for that pixel. If the difference between these two values does not exceed the threshold, the new value is thrown away and the old value is retained. If the new value is to be used (i.e., the difference between the old and new values for the pixel exceeds the threshold), then the new value is mixed into the old value based upon the mix setting. A low mix setting means that the new value will affect the old value only slightly. A large mix setting means the new value will greatly affect the old value, perhaps even replacing it completely.

You will find that convolutions affect the image data very severely. Therefore, you will probably want to use a low value for the mix setting so as to mute the effect of the convolution. The setting shown in Figure 6.5, for example, is a reasonable value for the **Sharpen** matrix. Figure 6.6 shows a ray tracing by Allen Hastings called "Hallway." The left half of the image has been sharpened using the Convolve operator, the right half has not.

As you would expect, 5 x 5 convolutions take longer to execute than 3 x 3 convolutions.

6.8 Crop_Image

The Crop_Image operator can be used to specify a rectangular region within an image which will be preserved. All data outside the selected region will be discarded.

NOTE Usage of this operator from the user interface is no longer suggested. Instead, use the Crop_Visual operator which provides you with an on-screen representation of the crop operation, rather than the numeric-only user interface which this operator presents. This operator is retained for use through ARexx control where the visual capabilities of Crop_Visual will not be used. See Section 6.9

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Figure 6.6: The left half of this ray tracing has been sharpened using the Convolve operator.

for more information about the Crop_Visual operator

.

When the Crop_Image operator is selected, it will bring up a control panel (shown in Figure 6.7) into which you can specify the offset (**X Offset:** and **Y Offset:**) of the top left-hand corner of the rectangle to be preserved. You specify the **New Width:** and **New Height:** of the area to be preserved as well. Note that you cannot specify a negative offset nor can you specify a combination of offsets and width or height which would exceed the width or height of the original image.

Aside from the straightforward use of this operator for cropping, you can also use this operator to "focus" ADPro's color picking technology. See **Focused Color Picking** in Appendix B for more information.

6.9 Crop_Visual

This operator provides an on-screen WYSIWYG user interface to the cropping operation. The **Crop_Visual** control panel is shown in Figure 6.8.

The box representing the area of interest indicates where the cropping opera-

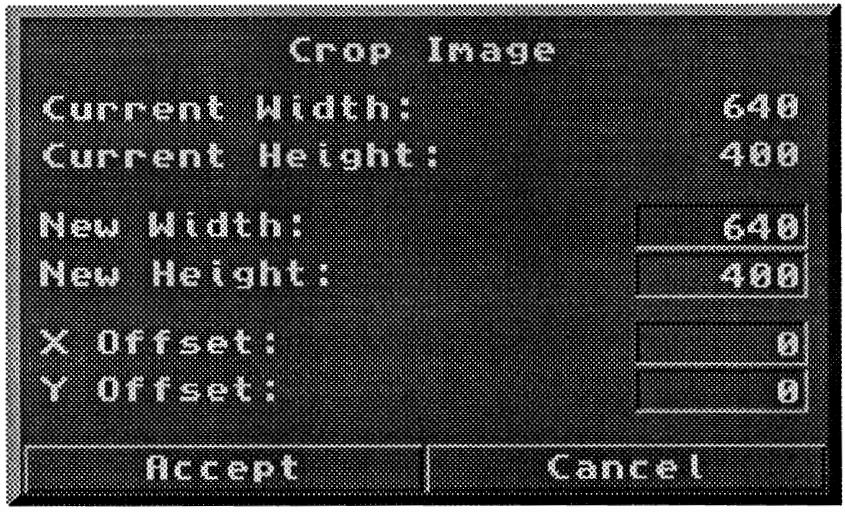


Figure 6.7: The Crop_Image operator control panel.

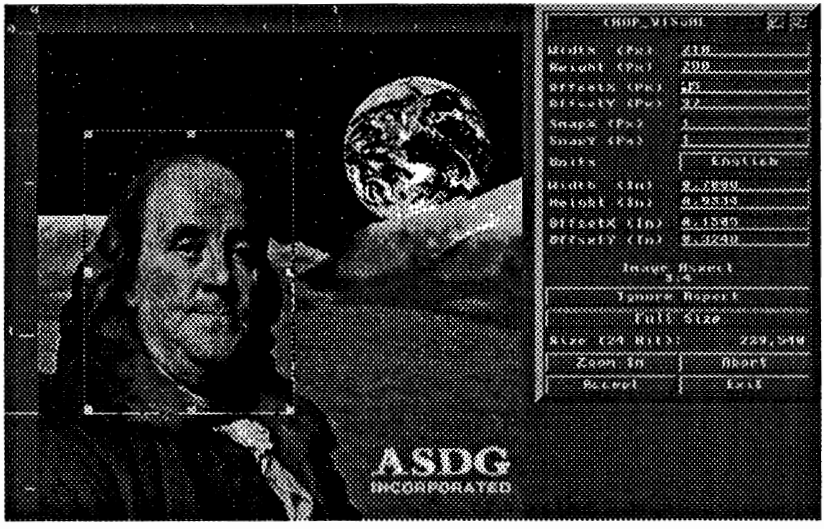


Figure 6.8: The Crop_Visual operator control screen.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

tion will take place. The portion of the image inside the box will be retained, while the area outside the box will be discarded. You can use the buttons marked **Full Size** and **Zoom In** to see finer or grosser detail. The **Accept** button will cause the cropping operation to actually take place. The **Cancel** button will leave the Crop_Visual operator without performing a crop.

Pressing **Shift + Return** on your keyboard is the same as hitting the **Accept** button. When the control panel is zipped, hitting the **z** key on your keyboard is the same as hitting the **Zoom In** button. Also, when the panel is zipped, hitting the **u** key on your keyboard will unzip the panel, making it full size again.

6.10 DCTV

The DCTV operator reformats the raw image data currently loaded in ADPro's memory into the format used by the DCTV display device (from Digital Creations). The reformatting overwrites the rendered image area in ADPro's memory but does not disturb the raw image area. If ADPro is not already set for 8 colors, the operator automatically shifts ADPro into Hi Res, 16 colors as required for DCTV. After the operator completes, it will display the resulting DCTV formatted image. This image cannot be scrolled from within ADPro.

If ADPro was already set to 8 colors, a 3 bit-plane DCTV image will be created instead of a 4 bit-plane version.

The DCTV operator takes its display size and type (i.e., interlace on or not, PAL or NTSC, etc.) from ADPro's current screen control settings.

You can save the DCTV version of the image by selecting the IFF saver. When saving a DCTV image, make sure you select the button marked *n* **Color Image** where *n* is either 8 or 16.

To decode a DCTV encoded image into 24-bit color data, read Section 6.29.

NOTE This operator requires and uses the **dctv.library** available from Digital Creations. If this library is not present in your **LIBS:** directory, you will get a message indicating a "fatal error in the operator". This library is included with this version of ADPro.

6.11 Define_Pxl_Aspect

ADPro adjusts the pixel aspect of the image as the image is scaled. After many scaling functions, the pixel aspect of the image may have little information left

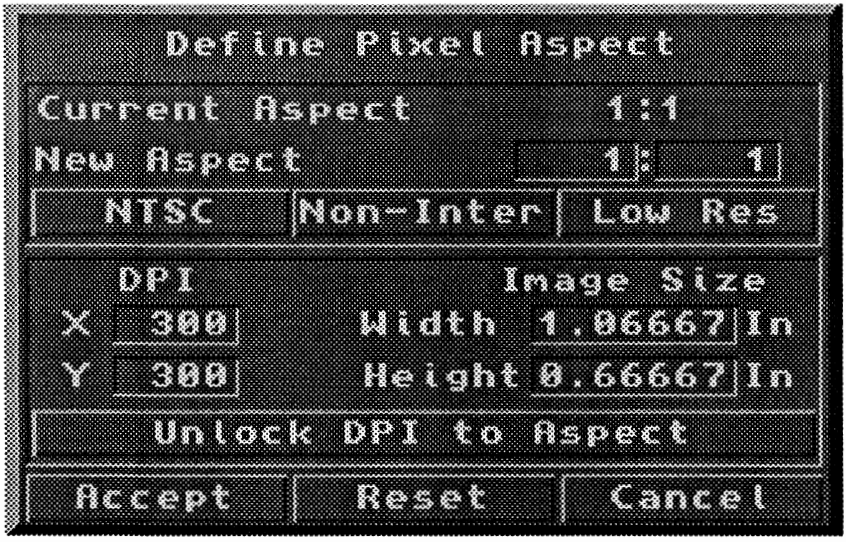


Figure 6.9: The Define_Pxl_Aspect operator control panel.

in it. For example, after scaling an image several times, you might arrive at an image that you wish to use as a low resolution interlaced picture. You don't care what the pixel aspect was before you started, but now that you have a picture you want to keep, you might want to redefine the pixel aspect to be exactly right for low resolution interlaced.

Also, many applications (such as desktop publishing) can make use of other resolution and sizing information which some file formats can maintain.

Using the Define_Pxl_Aspect operator, you can redefine the pixel aspect of the currently loaded image to be whatever value you wish. You can also redefine the "resolution" of the image so that programs such as desktop publishing packages will believe your image should be whatever size you want it to be.

Figure 6.9 shows the control panel for this operator. It displays the current pixel aspect and has two integer input fields which allow you to enter a new pixel aspect numerically. You can also specify a new pixel aspect by selecting the desired screen parameters from the buttons located just below these input fields.

Further, you can define an x and y resolution for your image. These values are maintained by ADPro and are actually used (in creating the rulers) in the operators.

Note that this operator does not actually modify any of the image data. It

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

simply redefines ADPro's internal idea of the image's pixel aspect and resolution. The new pixel aspect and resolution information will be saved with the image when the image is saved in a format which supports such information.

6.12 DeInterlace

The DeInterlace operator separates the currently loaded image into two images, stacked one upon the other. The top image is comprised of all of the even numbered scanlines (lines 0, 2, 4, etc.) from the original image. The bottom image is comprised of what were the odd numbered scanlines of the original image.

This operator is extremely useful for processing images grabbed with video digitizers such as the GVP IV24 and the Video Toaster (the Progressive Peripherals, Inc. FrameGrabber supports deinterlacing in hardware).

Such devices grab a full video frame which is comprised of two video fields, one sixtieth of a second apart (or one fiftieth of a second apart in non-NTSC regions). The time lag between fields can produce unwanted results particularly if the image being digitized contains rapidly moving elements. The DeInterlace operator can split the image data from the two fields apart so that either field may be processed separately.

To crop out either the top or bottom half of the image (i.e., the even or odd field), see the discussion on the Crop_Visual operator.

Note that the DeInterlace operator affects ADPro's impression of pixel aspect since the deinterlaced image would be properly shown on a screen half as high as the original image.

6.13 Displace_Pixel

The Displace_Pixel operator (shown in Figure 6.10) allows you scatter the pixels in an image. This, along with image compositing, can be used to create dramatic dissolves.

The **Radius** value defines the circular area in which any given pixel can be displaced from its current position.

The **Probability** value defines the likelihood that a given pixel will be displaced. This value can range between 1 and 100, where lower values mean that the pixel will less likely be to displaced and higher values mean a greater chance.

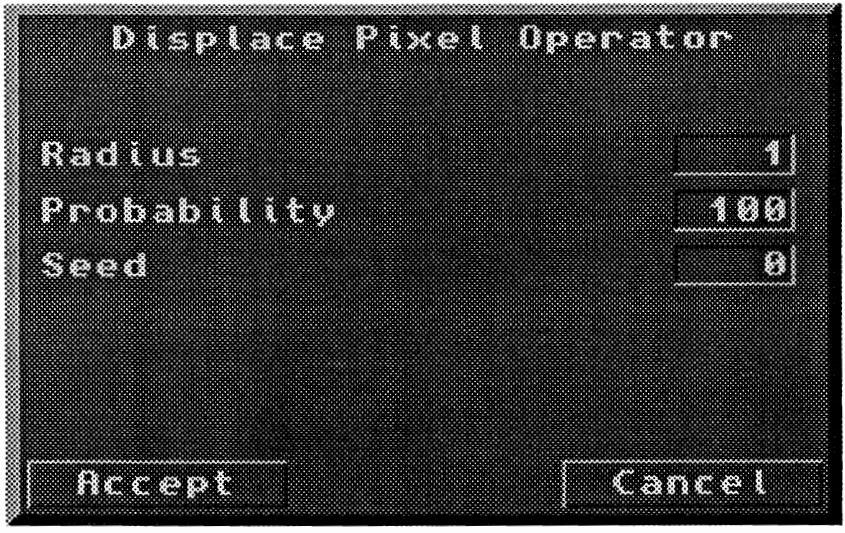


Figure 6.10: The Displace_Pixel operator control panel.

The **Seed** value "re-seeds" the randomizer, which determines in what direction a given pixel will be displaced. If you leave this value constant between invocations, you will get the same results. Changing it allows you to produce different results.

Press the **Accept** button to execute this operator on the image. Press the **Cancel** button to exit this operator without performing the operation.

6.14 Dynamic_Range

The Dynamic_Range operator performs two passes over any raw image data currently loaded in memory. During the first pass, the smallest and largest values contained in the raw image are found. Then, a control panel (shown in Figure 6.11) appears, displaying the values that were found in the first pass. If you select and accept a **New Maximum:** and **New Minimum:** value, a second pass over the raw image is performed, mapping all of the contained image data to lie between the selected maximum and minimum.

This operator can be used in many different ways. It can be used to expand or contract the dynamic range of an image (by increasing or decreasing the spread between the maximum and minimum values). It can also be used as a hybrid contrast/brightness manipulation.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

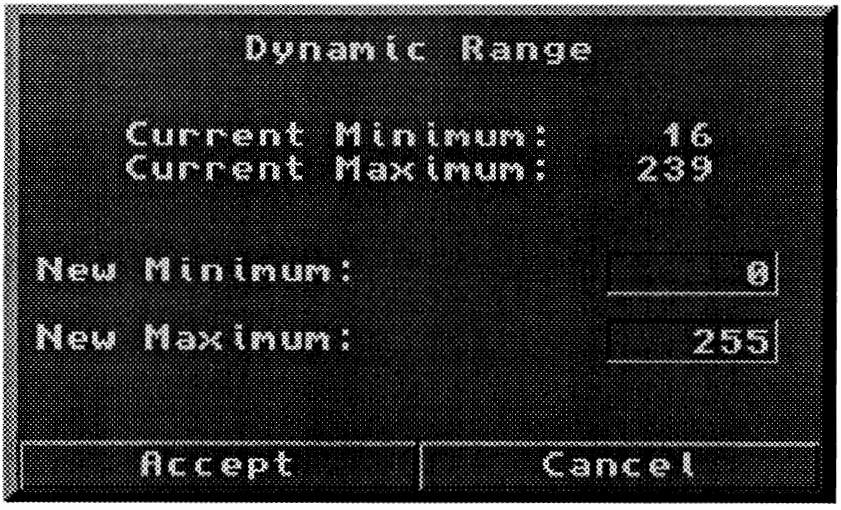


Figure 6.11: The Dynamic_Range operator control panel.

Since this operator allows colors to be uniformly compacted, video users can use this operator to prevent NTSC smear. This can be done by entering a new maximum value of approximately 208 rather than 255. Doing this causes the most intense color component picked for an Amiga palette to be 13 or less.

6.15 Gray_To_Color

The Gray_To_Color operator provides the logical inverse of the Color_To_Gray operator. Gray_To_Color reformats the internal data representation of raw grayscale image data into that of raw color image data. This feature is provided so that you can perform all image processing functions on both color and grayscale images.

When combined with image compositing, interesting artistic effects can be created. For example, a masked color image can be "pasted" into an otherwise completely grayscale image for a startling effect. See **Merging a Color Image Into a Grayscale Image** in Appendix B for more information.

6.16 Halve

The Halve operator is a highly optimized version of ADPro's scaling function designed to halve the width and height of the currently loaded image. This operator retains all of the precision of ADPro's scaling function, but executes approximately 13 times faster (on a 68030 processor). This operator is especially useful for CDTV developers who might frame grab thousands of images at video resolution but needs the results in quarter-screen size.

6.17 Horizontal_Flip

Many applications require the production of mirror images of the original data. Specifically, this is required by screen and heat-transfer printers. Selecting the Horizontal_Flip operator will horizontally flip the raw image data currently in memory.

Note that performing a horizontal flip followed by a vertical flip is the same as rotating the image 180 degrees. This can be used in conjunction with the **Orientation (Port / Land)** button to rotate images through 0, 90, 180, and 270 degrees. Note, as well, that you can also do these same rotations with the Rotate operator.

6.18 Intensity_Range

NOTE The Intensity_Range operator was developed by Steve Tibbett, and not ASDG, Incorporated. As such, you should contact Steve Tibbett's BBS at (613) 731-3419 for technical support. This module is included in the ADPro package under agreement with Steve Tibbett.

The Intensity_Range operator will build a graph showing you the relative number of pixels of varying intensities in the image, and let you bring the ends in, cutting out either the top, bottom, or both ends of the intensity range.

NOTE This operator requires at least AmigaOS 2.04 to function.

To use it, load or create a 24 bit image (gray and rendered images are not supported, but you can convert either of these into 24 bit with other ADPro operators), select the Intensity_Range operator and operate. Intensity_Range will build a histogram to show you 256 intensity levels (across the histogram) and the amount of the image at each intensity level. The graph is scaled so

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

that the highest concentration of intensity in the image is the height of the graph. The marks below the histogram are initially set to the top and bottom end of the intensity scale—when you move these and select **Accept**, the image will be remapped so that the area between the two marks will fill the entire intensity range. Any pixels whose intensity is above where you set the top mark are set to white, and pixels below where you set the bottom mark are set to black.

An example of the usage of this is taking a scanned image and running this operator on it: If the histogram shows that most of the image is grouped around one area, and that there are very few pixels outside that area, you might try bringing the ends in so that they encompass the bulk of the image. Cutting off an unused bit of the high or low end can sometimes improve an image quite a bit.

If an image is too dark, for example, bringing up the brightness will increase the brightness of the whole image—leaving you with no black in the image at all. Bringing the top end down to where the bulk of the image ends, will do a much better job of making a dark image seem brighter.

An interesting side effect is that you can tell what color resolution the image was created with. The histogram shows 256 intensities, but any standard Amiga image only has 4 significant bits—so you will only see a maximum of 16 spikes in the histogram. A scanned 24 bit image will usually show a smooth curve, and that is the kind of image that this operator works best with.

If you want to look for detail in the dark area of an image, then drag the top end down to near the bottom end and press **Accept**. Same for any other part of the image intensity range.

This operator works almost exactly like the Dynamic_Range operator that comes with ADPro, except that with Dynamic_Range you only know the highest and lowest intensities in the image, but not how the image really looks—one pixel at 255 and one at 0 will fool the Dynamic_Range operator.

This operator only works on 24bit image data.

6.19 Interlace

The Interlace operator facilitates the creation of interlaced images whose odd and even fields are composed of different pictures. This operator is the logical inverse of DeInterlace.

To interlace two different images together, use the BACKDROP loader to create a workspace tall enough to accommodate both images (i.e., twice as tall as either image alone). Then, using the compositing capability, load one image

into the top half of the backdrop, and the other image into the bottom half of the backdrop. Then execute this operator.

The resulting image will contain alternate scanlines from both of your original images.

Note that the Interlace operator affects ADPro's impression of pixel aspect since the interlaced image would be properly shown on a screen twice as high as the original images.

6.20 KillTemp

The KillTemp operator allows you to clear the image currently in ADPro's temporary buffer (placed there by the TEMP saver), whether or not an image exists in the buffer.

To determine if image data is currently in the temporary buffer, press the **About** button on the main control screen. If an asterisk (*) appears to the right of the **Buffer Size:** value, then the temporary buffer is not empty.

6.21 Line_Art

When 8 bit-plane grayscale data is available, selecting the Line_Art operator invokes a proprietary edge emphasis algorithm. The result is an image which contains the most prominent edges in the original image.

The production of quality line art from an arbitrary grayscale bit-map is quite difficult. It is important to begin with a high contrast original or to boost the contrast of the image before invoking the Line_Art operator.

Modifying the color controls (such as **Brightness**, **Contrast**, and **Gamma** correction) prior to executing the Line_Art operator will drastically affect the quality of the resulting line art. Increasing the brightness, contrast, or gamma correction of the image will also reduce the amount of clutter in the resulting line art. In general, increasing the contrast setting is usually a good step.

After the Line_Art operator has been executed, the color controls, such as brightness, contrast, and gamma correction, will only slightly affect the rendered image. Decreasing brightness or contrast will thicken the lines in the image. Increasing brightness, contrast, or gamma correction will thin out the lines in the rendered image.

When working with the Line_Art operator, we suggest that you save the original 8 bit-plane grayscale data to disk before executing the operator. In this

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

way, you can reload the original 8 bit-plane grayscale data and retry the command with different brightness, contrast, and gamma correction settings.

Note that executing the Line_Art operator will reduce the size of the available image data by 2 pixels in width and height.

6.22 Median_Filter

The Median_Filter operator computes the median color for each pixel in the image based upon itself and the 8 surrounding pixels. If the center pixel differs from the computed median by more than a specified threshold, the pixel is replaced with the computed median.

This operator can reduce noise or small unwanted artifacts in an image by replacing pixels which "seem not to fit in" with values which match the surrounding pixels more closely.

You specify a **Threshold** which determines how greatly a given pixel must differ from its surrounding pixels before it will be replaced. A lower threshold (which may range from 0 to 255) causes more pixels to be affected.

NOTE Each of the primary colors is considered separately. This can sometimes introduce artifacts rather than obscure them.

The Median_Filter operator allows a given threshold to be "pencil checked" using the **Test** button. By selecting this button, all computations will be made, but no changes will be made to the original data. The number of pixels which *would have been* affected will be reported in the **Pxls Affected:** text field. You can use this feature to fine tune your selection of **Threshold**.

Selecting the **Cancel** button will exit from the operator without modifying the data. Selecting the **Accept** button will apply the currently defined median filter operation to the raw data.

6.23 Negative

The Negative operator inverts any raw image data currently loaded in memory. For grayscale data, this produces what would commonly be called a black-and-white negative image of the original data. For color image data, each primary color is inverted separately.

This operator requires the presence of raw image data in memory.

6.24 OpalPaint

NOTE The OpalPaint operator was developed by Opal Technology, and not ASDG, Incorporated. As such, you should contact Opal Technology for technical support. This module is included in the ADPro package under agreement with Opal Technology.

Also, you must have at least version 1.1 of the OpalPaint program (not the ADPro operator) and enough memory to run the OpalPaint program (you might need to use the **MAXMEM** Tool Type). The latest versions of the OPALVISION software is available directly from Opal Technology / Centaur Development or their support BBS at (310) 793-7142.

Be sure you have the proper **assigns** applied before executing this operator.

The OpalPaint operator allows you to use the OpalPaint program (included with the OPALVISION display board) to touch-up or modify the raw image data in ADPro.

After selecting this operator and a short delay you will be presented with the OpalPaint program's screen, but with your raw ADPro image already loaded into the program. You can then use all the tools available in the OpalPaint program to edit the image.

Once you exit out of OpalPaint, and if you have modified the image, a requester will appear asking you if the changes should be preserved. If you accept these changes, the modified image will be reloaded into ADPro. Otherwise, your original image will be restored.

6.25 Polar_Mosaic

The Polar_Mosaic operator (shown in Figure 6.12) allows you to create a "tiled" image using concentric circles and "pie-style" cuts. Think of the mosaic circle as a pie that you can make radial (from the center point outward) slices, as well as concentric slices (tracks).

When the grayscale representation of your image is first being drawn on the screen, you can interrupt it by pressing the **Abort** button.

The center of the mosaic circle can be modified by either dragging the on-screen circle around the screen or typing the circle's center coordinates in the **Center X (Px)** and **Center Y (Px)** input fields. Notice that you get immediate feedback of your changes. You can center the circle horizontally, vertically, or both on the image by using the **Center X** and **Center Y** buttons.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

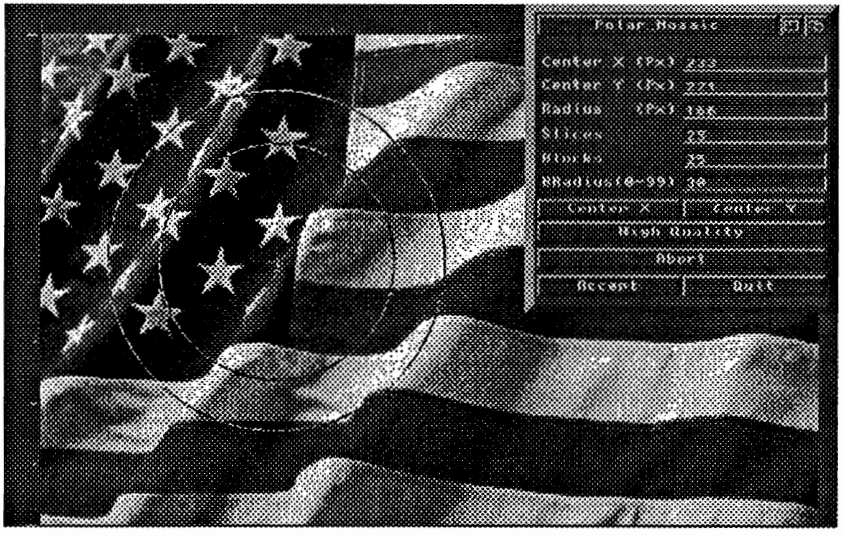


Figure 6.12: The Polar_Mosaic operator control panel.

The size of the mosaic circle is described by its radius and defined in the **Radius (Px)** input field. You can also click virtually anywhere on the screen to adjust the circle's size. The circle will expand to contract to the location of the mouse pointer.

The number of "pie slices" to be made in this circular area is defined by the value in the **Slices** input field.

The number of concentric "tracks" is defined in the **Tracks** input field. The more tracks you create, the more the original image will be recognizeable in the modified image, and vice versa.

The area near the outside edge of the circle can be "blurred" by specifying a value in the **BRadius(0-99)** input field. This value represents a percentage of the radius (starting from the outermost part of the circle going inward) to be blurred. A value of 0 means no blurring, while a value of 99 means that the entire circle will be blurred.

The quality of the effect can be controlled by toggling the quality rocker switch between **Fast** and **High Quality**. **Fast** works faster, but may not produce as precise an image as **High Quality**.

Press the **Accept** button to perform the operation, or **Quit** to return to the main screen without performing the operation.

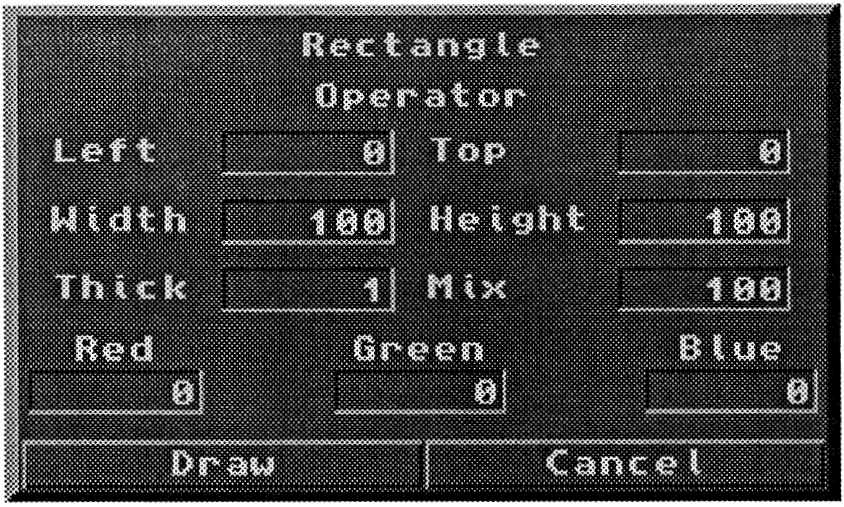


Figure 6.13: The Rectangle operator control panel.

6.26 Rectangle

The Rectangle operator can be used to draw filled and unfilled rectangles into a color or grayscale image, such as for borders around or drop shadows for rectangular regions. Selecting this operator causes the Rectangle control panel to appear. This control panel is shown in Figure 6.13.

NOTE Use of this operator from the user interface is no longer suggested. Instead, use the Rectangle_Visual operator which provides you with an on-screen representation of the operation, rather than the numeric-only user interface which this operator presents. This operator is retained for use through ARexx control where the visual capabilities of Rectangle_Visual will not be used. See Section 6.27 for more information about the Rectangle_Visual operator.

Using this control panel, you can specify the location of the upper left-hand corner (in the **Left** and **Top** input fields) of a rectangle as well as its **Width** and **Height**. You can also specify the thickness (**Thick**) of the rectangle. The default thickness is 1, which means that a 1 pixel thick rectangle will be drawn. Specifying a larger value means that a thicker rectangle will be drawn. Since you specify the *outside* dimensions of the rectangle, any additional thickness will be drawn within the inside of specified region. A thickness of -1 causes a filled rectangle to be drawn.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

You may also specify a **Mix** value to be used in the drawing computation. A mix value of 100 is the default, which indicates that drawn pixels shall completely obscure any pixels from the original image which they overlap.

Lastly, you may specify the **Red**, **Green**, and **Blue** content of the color to be used in drawing the rectangle. When drawing a rectangle in a grayscale bit map, only the red value is used to determine the shade of the rectangle. Colors are specified in the range of 0 to 255.

6.27 Rectangle_Visual

This operator provides an on-screen WYSIWYG user interface to the rectangle operation. We suggest that you use this operator rather than the Rectangle operator when working with ADPro from its user interface. However, when working from ARexx, we suggest that you continue to use the Rectangle operator since its small size will allow it to load from disk more quickly than Rectangle_Visual. Also, from ARexx, the visual benefits of Rectangle_Visual are negated.

The Rectangle_Visual control panel is shown in Figures 6.14 and 6.15. Figure 6.14 shows how a rectangle which is to be completely filled is represented on-screen. When the rectangle is drawn with an "X" through it, it means that the entire interior of the rectangle will be filled. Figure 6.15 shows a rectangle which will not be completely filled. Instead, it will be drawn with a thickness of 20. In all cases, the handles on the outer rectangle are what you use to control the size and location of the rectangle.

You can set the thickness of the rectangle using the input field marked **Thickness**. If set to -1, the rectangle will be filled, regardless of its size. If set to a number larger than -1, the rectangle will be drawn with the appropriate thickness. Note that having a thickness means that, at some sizes, the box will be completely filled. Suppose you defined a thickness of 50. Clearly, if the box were less than 50 pixels high or wide, the box would be completely filled. However, if the box were 500 pixels wide and tall, it would not be completely filled.

You can specify how much pixels being drawn as part of a rectangle will replace the background using the input field marked **Mix**. You can specify the color of the rectangle to be drawn using the input fields marked **R**, **G**, and **B**.

Selecting the **Accept** button will cause the currently specified rectangle to be drawn. When the drawing is complete, the screen will be updated and you can specify another rectangle. To exit the operator, press the **Cancel** button.

Pressing **Shift + Return** on your keyboard is the same as hitting the **Accept** button. When the control panel is zipped (in its "iconified" form), pressing the

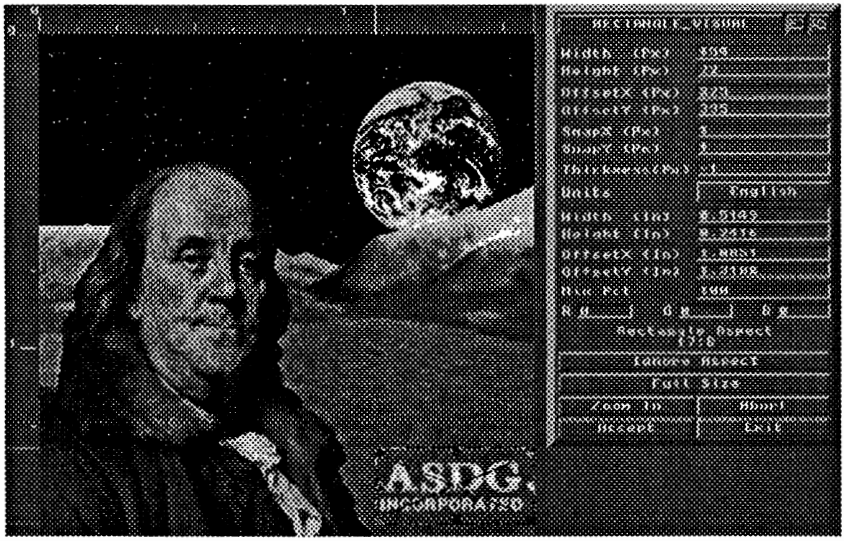


Figure 6.14: The Rectangle_Visual control screen and panel showing how a completely filled rectangle is represented.

z key on your keyboard is the same as selecting the **Zoom In** button. Also, when the panel is zipped, pressing the u key will unzip the panel (make it full size again).

6.28 Rem_Isolated_Pxls

The Rem_Isolated_Pxls (RIP or Remove Isolated Pixels) operator is used to "clean up" an already rendered image. It does so by examining each pixel in the rendered image data in succession. If the surrounding 8 pixels are the same color and the center pixel is a different color, the center pixel is replaced with the surrounding color.

We have found that Rem_Isolated_Pxls can "clean up" a majority of unwanted artifacts when rendering images with a small number of colors, such as a colored logo. Rem_Isolated_Pxls is less useful, though it can produce some artistic results, when used on color scans or other true color bitmaps.

In order to use Rem_Isolated_Pxls, you must have first rendered an image into 2, 4, 8, 16, 32, 64, 128, or 256 colors or grayscales. Rem_Isolated_Pxls does not operate on HAM images.

After rendering the image, selecting the Rem_Isolated_Pxls operator and de-

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

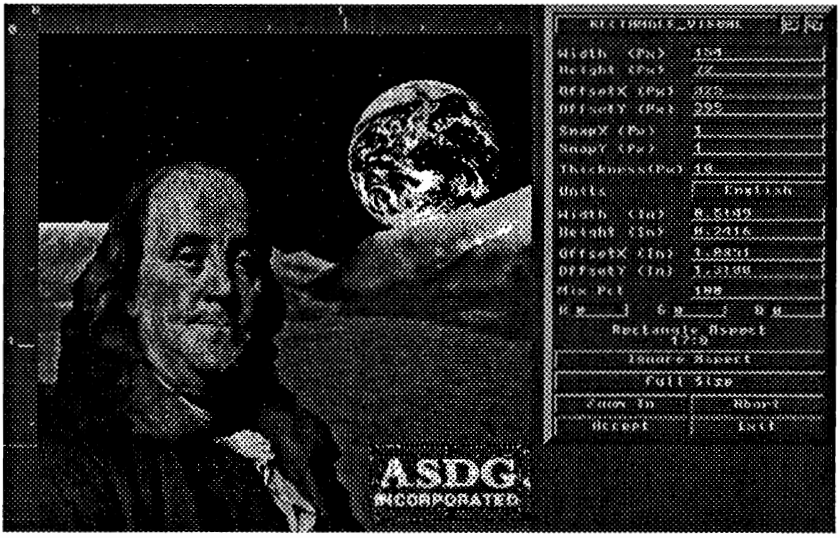


Figure 6.15: The Rectangle_Visual control screen and panel showing how a rectangle with a non-zero thickness (but not completely filled) is represented.

pressing the **Execute Op** button causes the operation to take place.

Rem_Isolated_Pxls operates directly on the rendered image data. Therefore, aborting a RIP in progress will result in a partially RIPped image.

6.29 Rendered_To_Raw

The **Rendered_To_Raw** operator converts the rendered data currently held within ADPro's memory back into the raw data it represents. In doing so, it replaces the raw data which was previously held in ADPro's memory. This operator makes it possible, for example, to save rendered data in file formats which only support raw data. Or, to send rendered data to devices whose savers only support raw data.

The operator can also be used for special effects. For example, if you wish to composite a true color object over a 16 color backdrop, you can render the 16 color backdrop, invoke this operator to convert the rendered 16 color data back into 24 bit-plane raw data, then composite in the true color foreground object.

Also, if you have the latest **dctv.library** (at least version 3.48) in your **LIBS:**

Figure 6.16: The Roll operator control panel.

directory, you can use this operator to decode DCTV images back to raw image data. Simply load in a DCTV encoded image using the IFF loader, then execute this operator. Rendered_To_Raw will detect that the currently loaded image is a DCTV image and will properly decode it back into raw data. The version of the `dctv.library` that allows for this conversion is included with this version of ADPro.

This operator can fail only if there is not enough memory available to hold the raw version of the rendered memory currently in ADPro's memory.

6.30 Roll

The Roll operator (as shown in Figure 6.16) will allow you to create roll effects from within ADPro. This operator is useful in conjunction with the image composition facilities in ADPro to create animations where one picture shifts off screen while another image shifts on screen.

The **Wrap** button will allow you to specify whether or not the image will wrap as it rolls. If **Wrap** is specified, any portion of the image moved off the screen on one side, will appear in the vacated space on the other side. If **No_Wrap** is specified, the vacated space is filled with black.

The Roll control panel contains eight direction arrows. Each arrow represents the direction in which the picture will be rolled.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

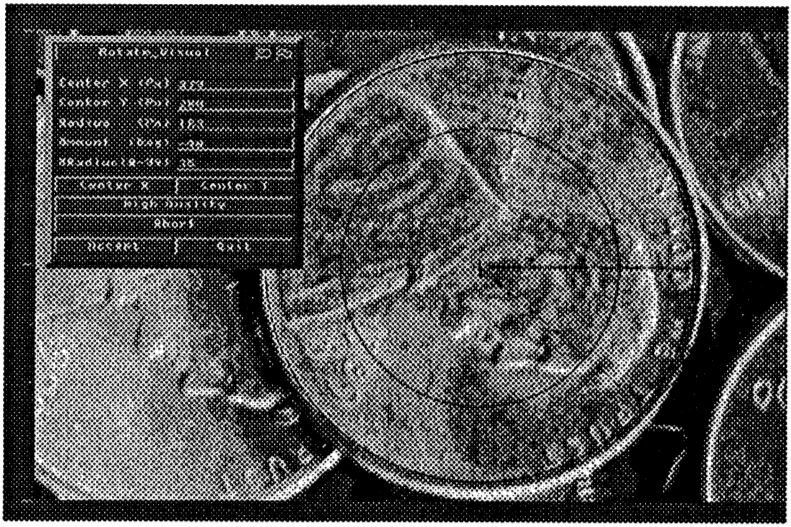


Figure 6.17: The Rotate operator control screen.

The **Pixels** input field tells the Roll operator by how many pixels you would like the image rolled. If you set this value to 100, the image will be rolled 100 pixels in the specified direction.

6.31 Rotate

Rotate's meter window will appear atop ADPro's screen & not its own.

With the Rotate operator, you can rotate circular areas of your image. The direction, center, amount, and quality of rotation, as well as the amount of blurring used at the edge of the circle, can be controlled.

Select the Rotate operator from the ADPro operator list. After selecting the operator, a preview screen should appear with the Rotate operator control panel on top of a grayscale representation of the currently loaded image.

The Rotate Circle visually describes the circular portion of your image that will be rotated. As you can see in Figure 6.18, the Rotate Circle is made up of a thin circle with a crosshair at its center, a smaller circle within this, a horizontal radius line in the right half of the circle, and a dashed line extended from the center of the circle.

Although values can be entered into the control panel's input fields, the more intuitive way of specifying the location of the rotate area on the image is to move the center crosshair on the Rotate Circle. Moving this crosshair moves

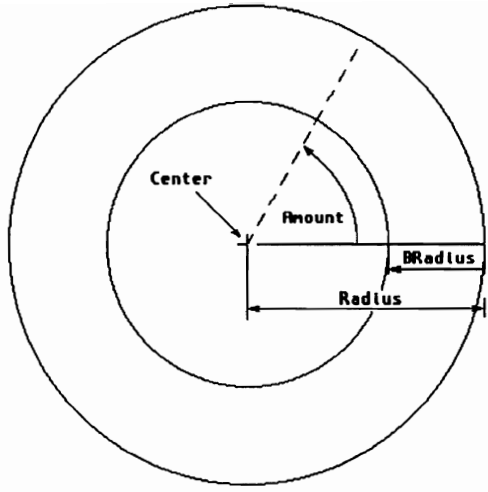


Figure 6.18: Components of the Rotate Circle.

the Circle as a whole around the image. To do this, place the mouse pointer above the crosshair, hold down the left mouse button, move the mouse so that the crosshair moves to the new location, and release the hold of the mouse button. Notice that when you move it around the image, the values in the **Center X (Px)** and **Center Y (Px)** input fields change as well. This gives you real-time feedback of your changes.

To center the Rotate Circle horizontally on your image, select the **Center X** button. To center it vertically, select the **Center Y** button.

Now that you've selected the center of your rotate area, you will want to specify the radius of this Circle. If you click the left mouse button while the pointer is anywhere but on the crosshair, the radius of the circle will snap to the new location. The actual radius (in pixels) is shown in the panel's **Radius (Px)** input field.

To define how much rotation will be done, you will have to enter a value in the **Amount (Deg)** input field. The current amount of rotation is shown on the Rotate Circle as a dashed line extending from the center of the Circle. An **Amount (Deg)** of 0 (no rotation) places the dashed line directly on top of the horizontal radius line. Positive **Amount (Deg)**s move the dashed line in a counter-clockwise direction. Negative **Amount (Deg)**s move it clockwise. Note that you cannot modify this dashed line's position with the mouse pointer.

The amount (percent) of blurring at the edge of the Rotate Circle can be

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

controlled by the value specified in the **BRadius(0-99)** input field. This is represented as the innermost circle within the Rotate Circle. A value of **0** produces no blurring, whereas a value of **99** produces blurring across the entire radius of the Circle. The position of this inside circle, as well, cannot be modified with the mouse pointer.

The speed (and correspondingly, the quality) of the rotate operation can be controlled with the **Quality** button. Clicking on the button will toggle the type of quality between **Fast** (not the best quality, but faster to complete) and **High Quality** (high quality, but slower to complete).

To abort the display of the current image, select the **Abort** button. Only a portion of the entire image may be shown on screen, but you can still move the Rotate Circle almost anywhere on the image.

To accept the current Rotate Circle values and begin the operation, select the **Accept** button. A meter window will appear on the Rotate control screen, showing the progress of the operation. To abort the operation at any time, select the **Abort** button.

If you have finished modifying your image for this session and want to return to the ADPro screen, select the **Quit** button.

6.32 Saturation

This operator (whose control panel is shown in Figure 6.19) allows you to adjust the amount of color saturation (using the **Percent** slider and input field) in an image. If a negative value is given, this operator will dull the colors while a positive value will make the colors more intense. Specifying a value of zero will cause no change to the image.

Specifically, this operator will either move colors closer to or further away from gray depending upon the value you specify. The operator can base its computation on three different color models.

YUV In the YUV model, pixels will be moved closer to or further away from gray as defined by the Y or Luminance component of each pixel.

HSV In the HSV model, pixels will be moved closer to or further away from gray as defined by the V or Value component of each pixel.

HLS In the HLS model, pixels will be moved closer to or further away from gray as defined by the L or Lightness component of each pixel.

Note that since all computations are made within one of three color models, specifying the same adjustment will produce different results based upon which color model was set at the time of execution.

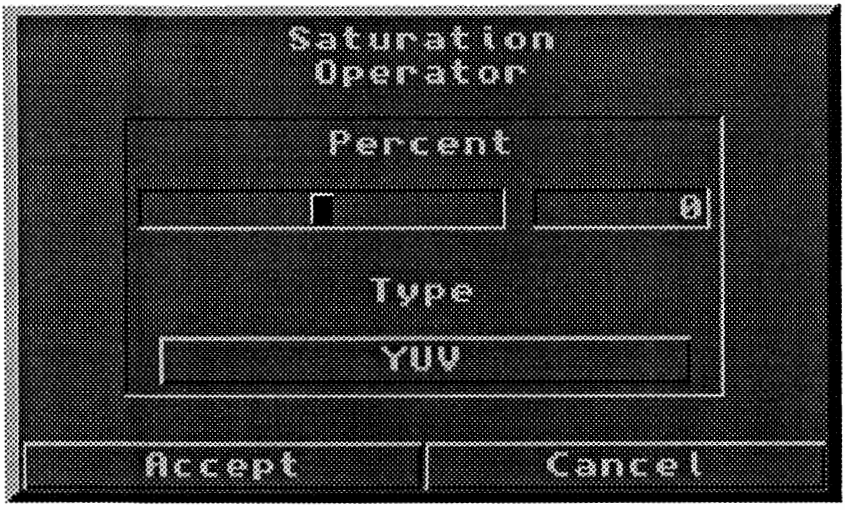


Figure 6.19: The Saturation operator control panel.

6.33 Scale

When 8 bit-plane gray or 24 bit-plane color data is present in memory, you may digitally reduce or enlarge the width or height (or both) of the image. This is useful in a number of ways. For example:

- The individual pixels in an NTSC screen are not square. In fact, they are higher than they are wide in low resolution, non-interlaced displays. However, the pixels produced by many image sources such as scanners and 3D modelling programs are perfectly square. This mismatch in aspect ratio can result in NTSC images which appear vertically stretched.

  By reducing the height of an image to approximately 86 percent of its original size, you can produce images with an accurate aspect ratio for NTSC screens.[1]
- Various artistic effects can be accomplished by reducing the width and height individually.
- You can create low resolution interlaced images easily by reducing the width alone by 50 percent and then enabling interlaced mode.

[1]Since the actual aspect ratio of a given screen will vary from monitor to monitor, the figure of reducing an image's height by 14 percent is *only approximate*. Also, PAL video is much closer to square than NTSC. Therefore, PAL screens may not require adjustment at all.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

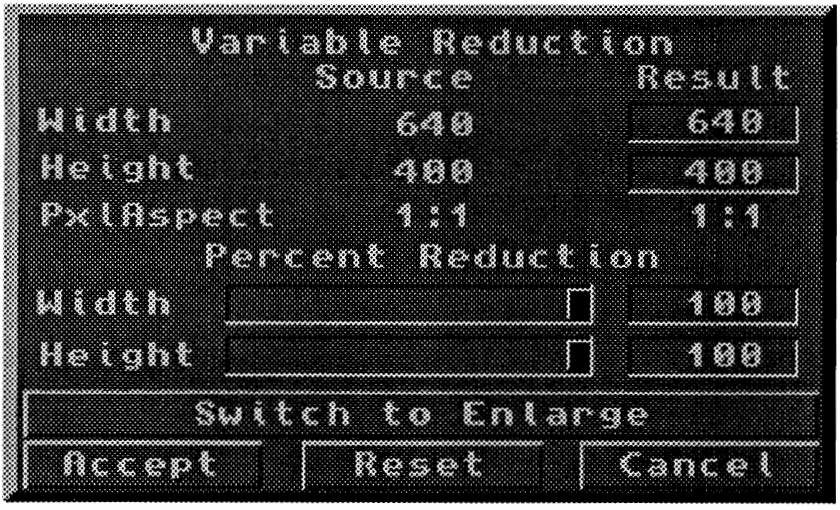


Figure 6.20: The Scale operator control panel.

- You can create high resolution non-interlaced images (such as in the preparation of icons for the Workbench) by reducing the height alone by 50 percent.
- You can cause an image to fit into any desired standard screen size. You can preserve the image's aspect ratio by reducing both the width and height by the same amount. Or, you can reduce the width and height by different amounts.
- You can produce a special effect called "pixelization" or "posterization" by significantly reducing the image and then significantly enlarging it.

Selecting the Scale operator when raw data is available in memory will cause the scaling control panel (shown in Figure 6.20) to appear.

You can switch between reduction and enlargement by selecting the button marked **Switch to Enlarge**. At any one time, you can either reduce or enlarge, but not both. When in the enlargement mode, the button which will switch you to reduction will be marked **Switch to Reduce**.

Using this control panel, you can specify a reduction or enlargement in two ways:

1. You can specify the number of pixels to be the resulting width and/or height using the supplied integer input fields.
2. You can specify the percent of reduction (or enlargement) of the width and/or height using the slider or input field.

When using the keyboard to enter values, the **Return** key will toggle between the width and height percentage of change or the new width and height in pixels depending upon which pair of input fields is active. **Alt + Return** will toggle between the pairs of input fields.

To reset the values to what they were when the control panel first appeared, you can select the **Reset** button. To ignore any scaling you might have specified, you can select the **Cancel** button.

Note that using the sliders, you can only specify reductions down to 25 percent of original size, or enlargements up to 400 percent of original size. However, by entering the scaling amounts manually, you can define reductions to less than 25 percent of original size, or enlargements to larger than 400 percent of original size.

To accept a set of reduction or enlargement values, simply select the **Accept** button or depress **Shift + Return** from the keyboard. *Unlike other operators, scaling operations are initiated immediately after acceptance.*

Quality digital scaling requires massive computation. The computation time of ADPro's scaling facility is proportional to the size of the resulting image. Also note that scaling permanently alters the raw data in memory. After an image has been scaled, you would have to reload from disk in order to get back the unscaled data.

6.33.1 Pixel Aspect

The IFF format supports the storing of a pixel aspect ratio with the file. When ADPro loads an IFF-ILBM file containing this pixel aspect ratio specification, it presents this information in the Scale control panel as shown in Figure 6.20. The pixel aspect defaults to 1:1 (pronounced "1 to 1") when ADPro loads a file which does not contain any pixel aspect information.

ADPro maintains the pixel aspect information as scaling operations take place. The aspect shown in the **Source** column in Figure 6.20 represents the current pixel aspect. The aspect shown in the **Result** column represents the pixel aspect which would result from the currently defined scaling operation if it were to be executed.

This information can be used to your advantage especially if you know that the device this picture is intended for display (or printing) has a specific pixel aspect. By adjusting the scaling controls so that the pixel aspect shown in the **Result** column matches the intended pixel aspect of the device, the aspect of objects within the image will be preserved.

For more information about pixel aspect, please see Section 2.7.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

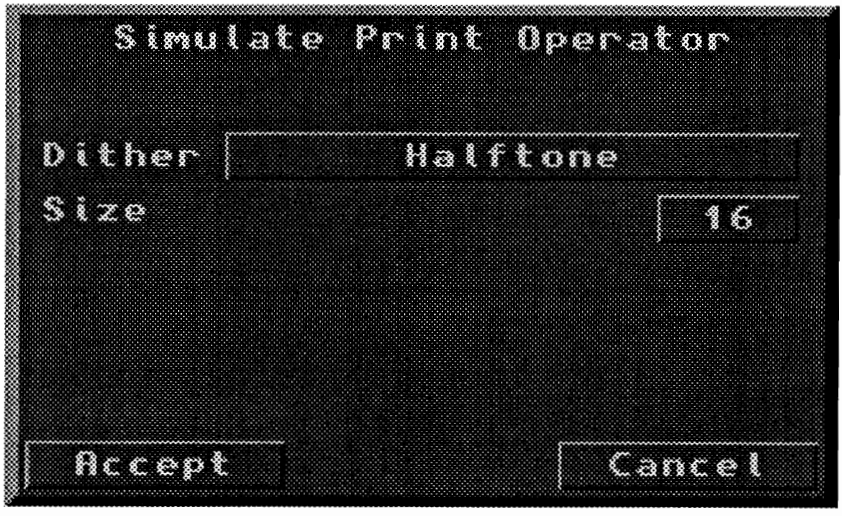


Figure 6.21: The Sim_Print operator control panel.

6.34 Sim_Print

The Sim_Print (Simulate Print) operator allows you to simulate the printing process, using the current raw image in ADPro's memory as its printing "surface." Specifically, an image that would have normally been dithered (to produce higher color fidelity) for paper output would itself be modified with the dither. Unlike other operators, the processing will not affect the original raw image data—the rendered image will be the simulated print.

As shown in the control panel (Figure 6.21), you have control over the **Dither** and **Size**. The **Dither** rocker switch lets you select the dithering method to use on the image. You can select from one of "Halftone," "Ordered," "VertiCalLine," "HorizontalLine," "FwdDiagonalLine," "BwdDiagonalLine," "FwdBrick," or "BwdBrick."

The **Size** rocker switch lets you select the size of the dither mask to use. You can choose from one of 2, 4, 8, or 16.

For a complete description of these dithering methods and mask sizes, see the PREFPRINTER saver section.

The **Accept** button will perform the simulated print on your image. The **Cancel** button will let you exit this control panel without executing this operator.

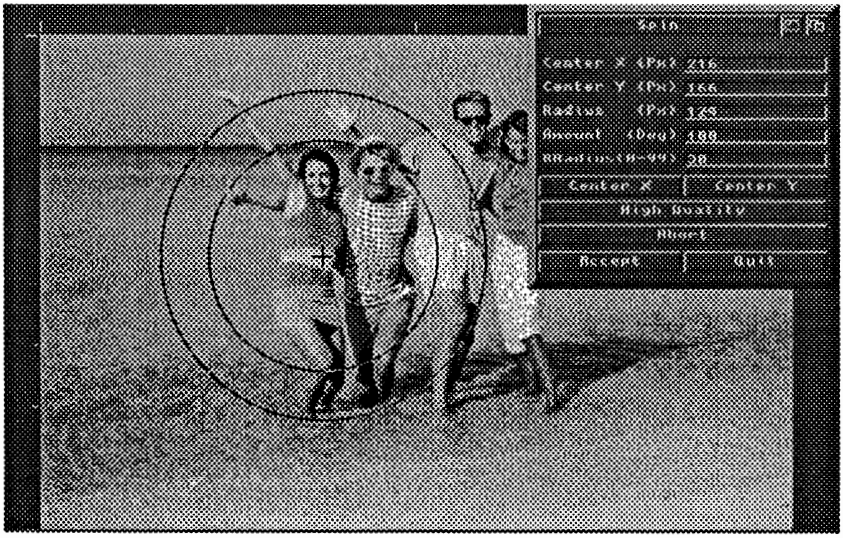


Figure 6.22: The Spin operator control panel.

6.35 Spin

The Spin operator (shown in Figure 6.22) allows you to rotate all the pixels within a circular area about the center point, based upon the intensity (brightness) of each pixel. The higher the intensity the more it will be displaced, and vice versa.

When the grayscale representation of your image is first being drawn on the screen, you can interrupt it by pressing the **Abort** button.

The center of the spin circle can be modified by either dragging the on-screen circle around the screen or typing the circle's center coordinates in the **Center X (Px)** and **Center Y (Px)** input fields. Notice that you get immediate feedback of your changes. You can center the circle horizontally, vertically, or both on the image by using the **Center X** and **Center Y** buttons.

The size of the spin circle is described by its radius and defined in the **Radius (Px)** input field. You can also click virtually anywhere on the screen to adjust the circle's size. The circle will expand to contract to the location of the mouse pointer.

The amount of "spinning" or rotation about the center point is controlled by the value in the **Amount (Deg)** input field. Positive values will rotate the pixel counter-clockwise, while negative values will rotate it clockwise.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The area near the outside edge of the circle can be "blurred" by specifying a value in the **BRadius(0-99)** input field. This value represents a percentage of the radius (starting from the outermost part of the circle going inward) to be blurred. A value of 0 means no blurring, while a value of 99 means that the entire circle will be blurred.

The quality of the effect can be controlled by toggling the quality rocker switch between **Fast** and **High Quality**. **Fast** works faster, but may not produce as precise an image as **High Quality**.

Press the **Accept** button to modify the image, or **Quit** to return to the main screen without performing the operation.

6.36 Text_Visual

The Text_Visual operator allows you to merge text into the ADPro image buffer using a WYSIWYG user-interface. The text can be written using several special effects including: embossing, tinting, and anti-aliasing. And, if you are working under Workbench 2.0 or later, you can use scalable (CompuGraphic) fonts to create a font of any point size you desire.

When you execute the Text_Visual operator, a grayscale representation of the ADPro image buffer is displayed on your screen, along with the operator's control panel. This panel can be used to enter your text string (or strings), choose a font type and size, control the placement of the text, and define which special effects will be used to draw the text.

The Text_Visual operator's main control panel (shown in Figure 6.23) is laid out into two sections. The buttons and input fields on the left hand side of the panel are concerned with choosing and placing your text string on the screen, whereas the controls on the right hand side of the panel are used to determine how the text is drawn into ADPro's image memory.

The buttons and input fields presented in the main control panel are described below.

The input field (labelled as **String:**) in the upper left hand corner of the control panel is where you enter the string you want to appear on the screen. For instance, let's say you wanted to place the phrase, `Hi there!` on the screen. Click in the **String:** input field and enter the string `Hi there!`. After you press the `Return` key, the text you entered will be rendered internally within the Text_Visual operator. For large strings and especially with large fonts, this may take a few moments.

When the text has been assembled within the Text_Visual operator's memory, it will be flashed on-screen. The on-screen representation of the text will have

Figure 6.23: The Text_Visual operator control screen and main panel.

been scaled to properly reflect the relative size of the text compared to the size of the image you are working on.

Note that you may have left the previous text positioned under the space now occupied by the control panel itself. You can tell when the text is finished being assembled for previewing when your mouse pointer assumes its default shape (after turning into the busy clock shape) again.

Directly below the **String:** input field are the closely related **Font**, **Type**, **Style**, and **Size** controls. Clicking on the **Font** button will bring up a list of all the available fonts on your system. If you have specifically requested to use scalable fonts, only the fonts whose name ends with **.otag** are allowed to be selected (since only these are outline fonts). If you have selected to use bit-mapped fonts, these will be presented along with those scalable fonts which have predefined bit-mapped sizes.

The **Type** rocker switch allows you to toggle between bit-mapped and (should you have them) scalable fonts. The scalable font option is available only if you are running under AmigaOS 2.04 or later.

When switching to scalable fonts or when changing the size of a scalable font, there is often a lengthy pause while the operating system performs the calculations it needs to convert a scalable outline into a bit-map ready for use. Consult the scalable font documentation from Commodore concerning ways you can decrease the length of this pause (by manipulating the font cache size,

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

for example).

The **Style** button allows you to choose various attributes for the text you wish to use. These attributes include **Bold**, **Italic**, **Extended**, and **Underlined** styles. Combinations of these styles are also permitted.

Note that font styles are handled by the operating system, not by ADPro. Some styles may not work properly with some fonts, especially with outline fonts. Also note that changing the style of a scalable font will often mean that the operating system must take some time to build the new type face from its scalable version.

The **Size** button allows you to choose the point size for the font you're currently using. If you are using a bit-mapped font, the size you pick must be one of the pre-existing point sizes. If you are using a scalable font, you can pick any size between 4 and 127 points. Note that if a pre-built bit-mapped version of the scalable font you're using doesn't exist, the operating system will take some amount of time to create it.

The **Tracking:** input field allows you to control inter-character spacing (the space between individual characters in your text string). Tracking can be used to improve readability or for artistic purposes.

Experiment with how this affects the look of your string by typing in various tracking values (try comparing zero tracking with a tracking of ten, for example).

Positive values expand the text string, negative values move the characters closer together.

Assuming you've entered a string (and pressed the **Return** key) and chosen a font, you can now move the rendered string around the screen by grabbing it with the mouse (while holding down the left mouse button).

This brings us to the **X Offset:** and **Y Offset:** input fields. These input fields were designed to aid you in placing your text string on-screen.

The **X Offset:** and **Y Offset:** input fields each have three boxes to their right. The first box to the right of each input field displays the coordinates of the text string on-screen. You can directly enter the screen coordinates in these input fields to position your text on-screen or, as mentioned before, you can use the mouse to position your text. If you use the mouse to position your text, the input fields immediately to the right of **X Offset:** and **Y Offset:** will be updated continually as long as you have the left mouse button depressed.

Let's skip to the third box to the right of the **Offset** input fields. These rocker switches determine the point on your string from which the screen coordinates are derived. Think of these as defining the "handle" you use to position the text on-screen. The **X Handle** rocker switch allows you to choose from the

Left, **Center**, and **Right** of the string; the **Y Handle** rocker switch allows you to choose from the **Top**, **Baseline**, and **Bottom** of the string.

The ability to define where the "handles" are placed make it easy to create left or right justified text, and horizontally aligned or vertically centered text.

Finally, the rocker switches immediately to the right of the **Offset** input fields restrict the movement of the text on-screen to help you line things up. These rocker switches have two states: **Free** and **Locked.**

Setting one of the switches to **Locked** freezes the axis so that no more movement along that axis is allowed. As an example, you might use the mouse to position your text to the right height. Then, flip the **Y Offset** rocker switch to its **Locked** state. Then, if you press the **Center** button, the text will be centered based only upon the X axis (width) and not along the Y axis (height).

The **Center** button (at the lower-left hand corner of the Text_Visual operator control panel) allows you to immediately center your text string on-screen. The **X Offset** and **Y Offset** rocker switches determine how the string is centered. If both **Offset** rocker switches are set to **Free**, then the string is centered in both X and Y (at the dead center of the screen) axes. If one of the axes is **Locked**, the string will be centered along the unlocked axis.

We now move on to the rocker switches and input fields on the right hand side of the Text_Visual operator control panel. These controls describe the appearance of the text, and how it will be drawn into the existing image in memory.

The effect of all of these controls are additive, so a extremely large number of special effects are possible. Experimentation is essential and rewarding.

The **Blur:** setting is used to control how much smoothing (or anti-aliasing) will be applied to the edges of each character in the text string. By changing the blur value, you can control how jagged or fuzzy the edges of each character will appear. Valid values are between -1 and 16.

A value of -1 disables blurring, so the characters will be drawn exactly as they are defined. As in the Blur operator (see Section 6.2), a value of 0 gives maximal blurring. A value of 16 gives minimal blurring.

The **Emboss:** rocker switch gives you the ability to give each character a three dimensional appearance. You can make your text appear as though it was punched into (setting **IN**) or is protruding from (setting **OUT**) its background.

Or, you can specify the **Emboss:** to be from one of the eight cardinal compass directions (**N**, **NE**, **E**, **SE**, **S**, **SW**, **W**, or **NW**).

The input field to the right of the **Emboss:** rocker switch holds a number

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

which is used to control the "amount" of embossing that occurs. Valid values for this number are between 0-100. As this number grows larger, the text will appear more embossed, that is it will appear to stick out of (or into) the picture more.

Directly below the **Blur:** and **Emboss:** controls is the button which is used to control how your text is overlayed (or drawn) into the screen. Clicking on this button causes it to cycle through one of five possible values: **Mix**, **Lighten**, **Darken**, **Transparent**, and **Outline**.

Immediately to the right of this button is an input field which can hold a number between 0-100 (as with the **Emboss:** button). This field controls how intensely each of the above mentioned options will take effect.

- Selecting **Mix** causes your text (and the color you have chosen for it) to be mixed with the underlying ADPro bitmap. The value in the intensity input field controls how much color from the background shows through. A value of 0 causes the background to show through entirely; a value of 100 causes your text to completely overwrite the picture below it.
- Selecting **Lighten** causes ADPro to lighten the image beneath your text without changing that area's hue. Since **Lighten** takes its coloring information from the hues already found in the image, the values you specify in the **Color:** input fields are ignored.

  The **Intensity** input field controls how much lightening occurs. A value of 0 would do nothing (no text would be seen), and 100 will cause the greatest effect.
- Conversely, **Darken** causes ADPro to darken the image beneath your text. As with **Lighten**, the values you specify in the **Color:** input fields are ignored since the colors used in the darkening operation are those already contained in the image.
- The **Transparent** choice turns off all direct mixing of the text. Other drawing, such as that caused by the embossing effects, will still be visible.
- The **Outline** option causes only the outline of the text to be drawn. Again, the **Intensity** value controls how much of the background beneath the text shows through. In order for **Outline** to work, blurring must be enabled.

The **Color:** control area contains three input fields. This is where you enter the desired color for your text string. Color is input in the 24 bit RGB space; valid values are between 0 and 255.

Directly below the **Color:** input fields lie the **Tint:** and **Saturation:** input fields. These two fields are used to give technical users more control over the appearance of their text. Unlike most operators in ADPro, which work in RGB (Red, Green, Blue) color space, these two input fields affect the color

of your text in HSV (Hue, Saturation, Value) color space. While a detailed explanation of color theory is beyond the scope of this manual, here is a brief explanation of the function of these fields.

- The **Tint:** field is used to control how strongly the RGB color you specify in the control panel will tint the pixels beneath the text. At a setting of 100, the pixel beneath the text will take on the RGB color you specify, completely. At a value of 0, the pixels beneath the text will be left unchanged. This function changes the hue of the pixels, but leaves alone their saturation and value.
- The **Saturation:** field works in the following way: The operator will compute the color saturation of the RGB color you specify in the control panel. At a setting of 100, the pixels beneath the text will assume completely the saturation of your RGB color. At a setting of 0, the pixels beneath the text will be unchanged. This function changes the saturation of pixels, but leaves alone their hue and value.

Once you have all the settings defined, you can press the **Accept** button to actually overlay your text with the ADPro bitmap. ADPro draws a box around each character on-screen as it is being processed. By watching the progress of this outline box as it travels across your text, you can follow the progress of the text operation.

Once it has finished, you can exit the Text_Visual operator (by clicking on the **Exit** button), or you can set up another line of text.

Clearly, there are a considerable number of possible effects and combinations of effects. For this reason, the **Load Setup** and **Save Setup** buttons allow to store favored settings on disk and recall them when desired.

6.36.1 Example Usage

Load an image of your choice into ADPro. Next, select the Text_Visual operator. After a slight delay, your image will be displayed in grayscale, along with the Text_Visual operator's control panel.

Click in the **String:** input field and type in a string of your choice (for example, `Hello World!`). Next, click on the **Type:** rocker switch so that it says **Bitmapped**, if it doesn't say that already.

Afterwards, click on the **Font:** button. This will bring up a list of the available fonts on your system. Most Amiga's should have the `emerald.font` font installed. Choose this font, with a point size of 20. (If you do not have this font, choose another font which you do have.)

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

After this, click in the **Tracking:** input field. Enter a value of 20. Notice how this expands the text after you press the **Return** key on your keyboard.

Now move the mouse up to the text string. Experiment dragging the text around the screen. If both the **X Offset:** and **Y Offset** range of motion rocker switches are set **Free**, then you should be able to move the text around anywhere within the borders of your image.

Set the text down at some particular height. Click on the **Y Offset:** range of motion rocker switch so that it says **Locked**. Now press the **Center** button. Notice how the text moves so that it is horizontally centered on-screen but that its height on-screen did not change. This is because the **Y Offset:** range of motion was "locked."

Now go over to the **Color:** input fields. In this case, set the Red (**R**) and Green (**G**) values to 255, and the Blue (**B**) value to 0. This will cause the text to be rendered in yellow.

Set a value of 8 into the **Blur:** input field and click on the **Emboss:** rocker switch until it displays **NW** (Emboss the text from the NorthWest). In the input field immediately to the right of the box displaying **NW** enter a value of 80.

Click on the long button immediately below the **Blur:** and **Emboss:** controls. Repeatedly click on it to cycle through the various options until the **Mix** setting is displayed. Now enter a value of 70 for the mix level (the input field just to the right of the **Mix** rocker switch).

Select the **Accept** button. ADPro will draw a box around each character in the string as it is drawn into the picture. When the operator is finished, quit it by selecting the **Exit** button.

From the main ADPro control screen, select the **Execute** button to see your changes in full color.

6.37 Tile

The Tile operator allows you to specify a rectangular area which will be repeated throughout the currently defined bit-map to create a "wall paper" or tiled effect. You can specify a vertical or horizontal offset which will skew the tiling either vertically or horizontally.

NOTE Use of this operator from the user interface is no longer suggested. Instead, use the Tile_Visual operator which provides you with an on-screen representation of the operation, rather than the numeric-only user interface which this operator presents. This operator is

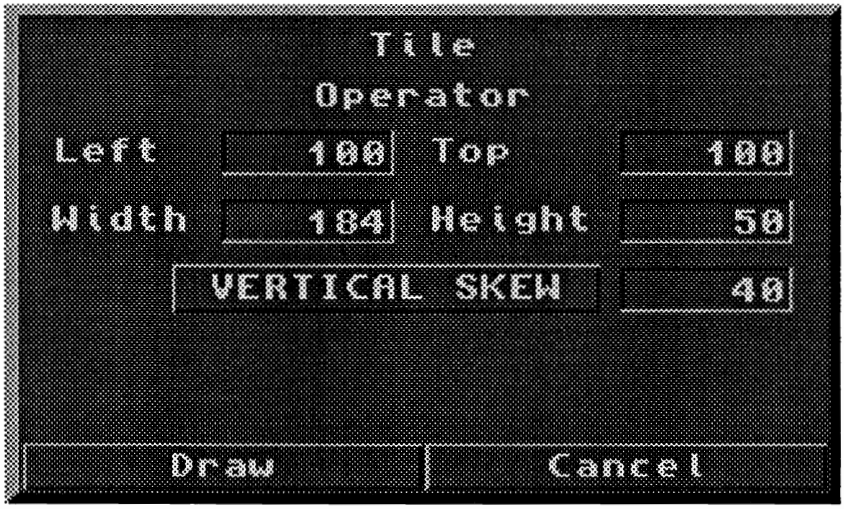


Figure 6.24: The Tile operator control panel.

retained for use through ARexx control where the visual capabilities of Tile_Visual will not be used. See Section 6.38 for more information about the Tile_Visual operator.

The Tile operator's control panel is shown in Figure 6.24. Specify the left edge (pixel column number) of the rectangle you wish to tile in the input field marked **Left**. Similarly, specify the top edge (pixel row number) of the rectangle you wish to tile in the field marked **Top**. Also specify the width and height of the rectangular area (in pixels) in the fields marked **Width** and **Height**, respectively.

The rocker switch marked **VERTICAL SKEW** in Figure 6.24 toggles between that setting and **HORIZONTAL SKEW**. The amount of skew is specified in the integer input field to the right of the rocker switch.

Selecting the **Draw** button causes the tiling operation to actually take place. Selecting **Cancel** exits the Tile operator without actually performing a tiling operation.

Figure 6.25 shows a 320 by 200 bit-map which has been tiled with a 32 by 20 pixel rectangle (shown in gray) with no skew. A no-skew tiling is characterized by the rigid grid pattern as Figure 6.25 shows. You can specify a no-skew tiling by setting the skew amount to 0 (the skew type may be set to either vertical or horizontal).

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

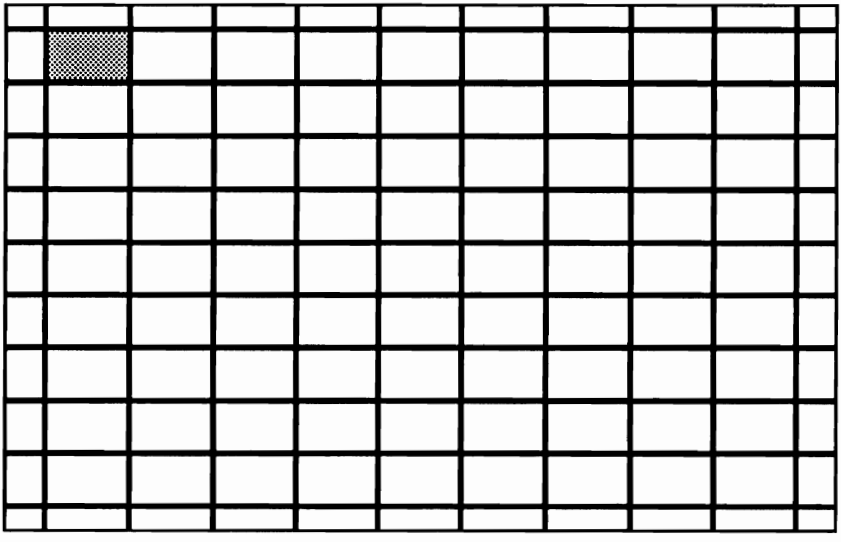


Figure 6.25: A tiling showing no skew.

Figure 6.26 shows a 320 by 200 bit-map which has been horizontally skewed using the same 32 by 20 pixel rectangle (shown in gray). A horizontal skew is one in which the tiled rectangles all line up horizontally.

Finally, Figure 6.27 shows the same 320 by 200 bit-map which has been vertically tiled by the same 32 by 20 pixel rectangle (shown in gray). A vertical skew is one in which the tiled rectangles all line up vertically.

Notice that the source of the tiling operation (shown in gray in Figures 6.25, 6.26, and 6.27) remains the same. This shows that you can repetitively execute the Tile operator over the same rectangle, varying the skew amount and type, as long as you keep the same width, height, left edge and top edge.

Also notice that the tiled area did not fit evenly into the total bit-map. That is, fractional portions of the tiled area can be found along the edges of the bit-map. This will necessarily happen when the width or height of the tiled area does not evenly divide into the width or height of the entire bit-map. You can also force this to happen to offsetting the placement of the tiled area from the top left corner of the bit-map.

In the examples shown in the preceeding figures, the tiled area was 32 by 20 pixels in size. The entire bit-map was 320 by 200 pixels in size. Clearly, by specifying a left edge and top edge of 0, the tiled area would evenly fit in the larger bit-map. This is shown in Figure 6.28. However, if you wish to force the tiled area to not fit evenly in the larger bit-map, you can reposition the source

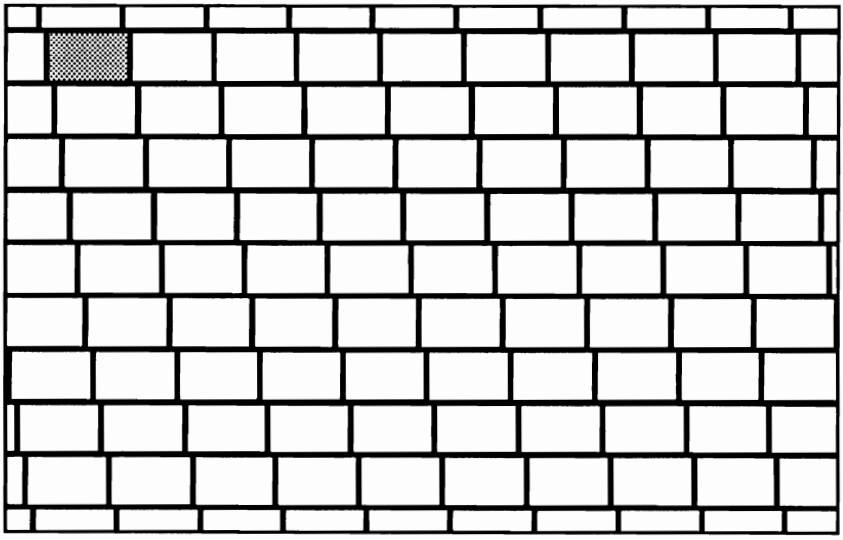


Figure 6.26: A tiling showing horizontal skew.

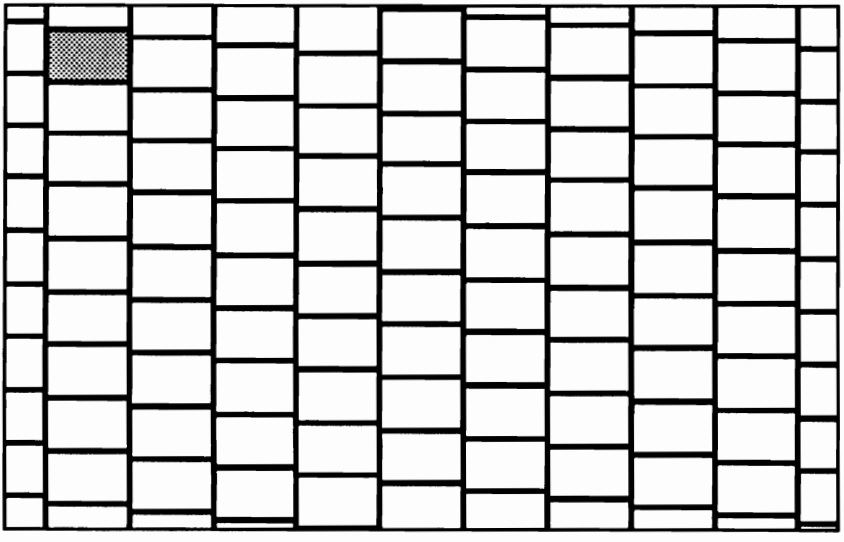


Figure 6.27: A tiling showing vertical skew.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

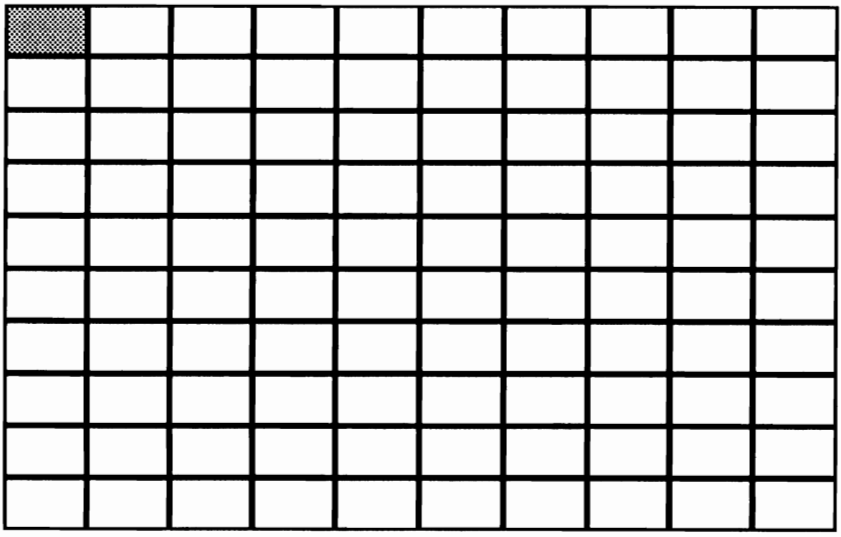


Figure 6.28: A tiling showing no skew in which the tiled area evenly fits in the larger bit-map.

rectangle as needed.

For example, if you wished to tile a 320 by 200 pixel bit-map with the 32 by 20 pixel rectangle starting at a left and top edge of 0, but wanted to force a fractional part along the borders of the bit-map, you might perform the following steps:

1. Using either the Crop_Image or Crop_Visual operator, isolate just the area you wish to tile from. Save this area to a temporary file.
2. Using the BACKDROP loader, create a bit-map of the desired size.
3. Using image compositing, load the area to be tiled from the temporary file at an offset which will yield the tiling you wish.
4. Tile the image, specifying a width and height of the area you are tiling from and left edge and top edge from where you loaded the area in the previous step.

6.38 Tile_Visual

This operator provides an on-screen WYSIWYG user interface (shown in Figure 6.29) to the tiling operation. We suggest that you use this operator rather

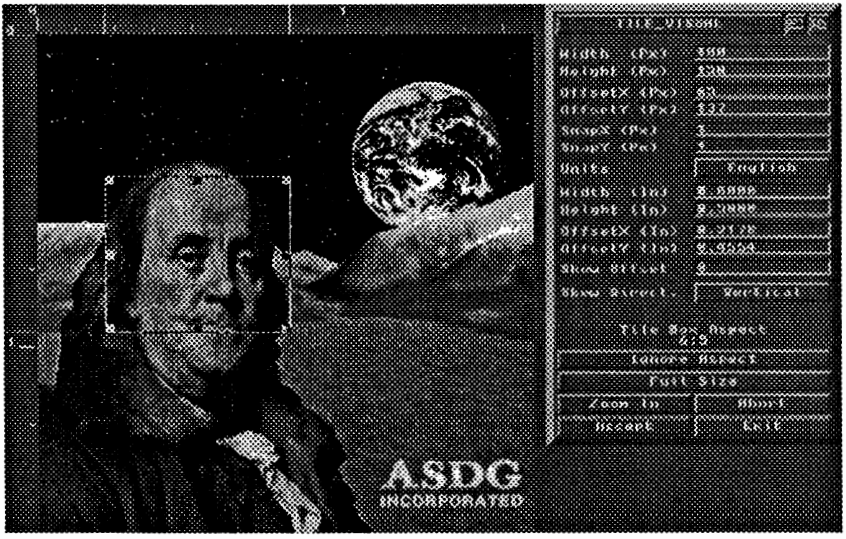


Figure 6.29: The Tile_Visual operator control screen and panel.

than the Tile operator when working with ADPro from its user interface. However, when working through ARexx control, we suggest that you continue to use the Tile operator since its small size will allow it to load from disk more quickly than Tile_Visual. Also, from ARexx, the visual benefits of Tile_Visual are negated.

The box representing the area-of-interest indicates where the bounds of the area which will be tiled. You can use the buttons marked **Full Size** and **Zoom In** to see finer or grosser detail. The **Accept** button will cause the tiling operation to actually take place. The **Exit** button will leave the Tile_Visual operator without performing a tiling operation.

Pressing **Shift + Return** on your keyboard is the same as selecting the **Accept** button. When the control panel is zipped (it is shown "iconified"), pressing the **z** key on your keyboard is the same as hitting the **Zoom In** button. Also, when the panel is zipped, pressing the **u** key will unzip the panel (make it full size again).

6.39 TPort_Controller

Selecting the Transport_Controller operator causes ADPro to look for a running copy of MicroIllusions' Transport Controller. If found, ADPro will com-

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

municate with the Transport Controller to record the Amiga displayable image currently in memory.

This operator requires the presence of rendered image data in Amiga displayable format. It will also tell you if it cannot locate a currently running Transport Controller.

Control will return to ADPro when the Transport Controller is through recording the image. The number of frames to be recorded is determined by the default value maintained by the Transport Controller. Direct control (from within ADPro) over the number of frames to be recorded is provided only via ARexx since the Transport Controller provides a user interface of its own.

6.40 Twirl

With the Twirl operator, you can twirl circular areas of your image. The direction, center, amount, and quality of twirl, as well as the amount of blurring used at the edge of the circle, can be controlled.

Select the Twirl operator from the ADPro operator list. Please refer to Section 2.4.1 for specific instructions.

After selecting the operator, a preview screen should appear with the Twirl operator control panel on top of a grayscale representation of the currently loaded image.

The Twirl Circle visually describes the circular portion of your image that will be "twirled". As you can see in Figure 6.31, the Twirl Circle is comprised of a thin circle with a crosshair at its center, a smaller circle within it, a horizontal radius line in the right half of the circle, and an dashed arc extending from the center of the circle out to the edge.

Although you can enter values into the control panel's input fields, the more intuitive way of specifying the location of this twirl area on the image is to move the center crosshair on the Twirl Circle. Moving this crosshair moves the Circle as a whole around the image. To do this, place the mouse pointer above the crosshair, hold down the left mouse button, move the mouse so that the crosshair moves to the new location, and release the hold of the mouse button. Notice that when you move it around the image, the values in the **Center X (Px)** and **Center Y (Px)** input fields change as well. This gives you real-time feedback of your changes.

To center the horizontal position of the Twirl Circle, select the **Center X** button. To center the vertical position, select the **Center Y** button.

Now that you've selected the center of your twirl area, you will want to specify

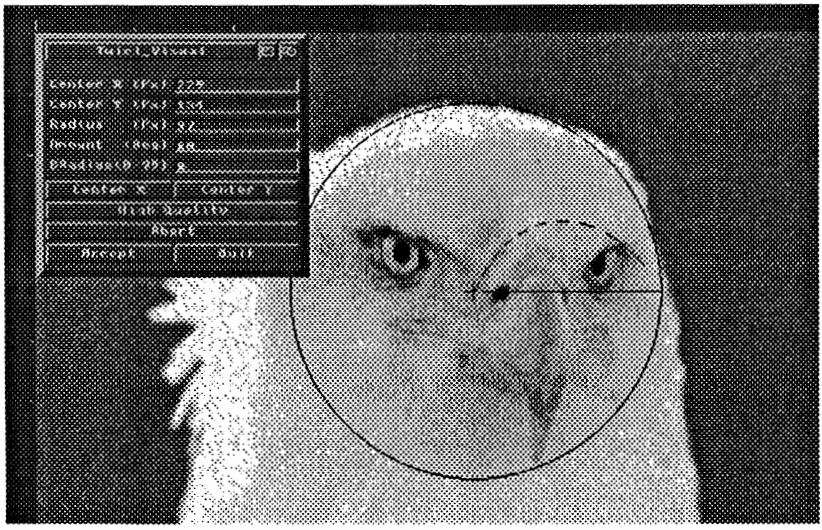


Figure 6.30: The Twirl operator control screen.

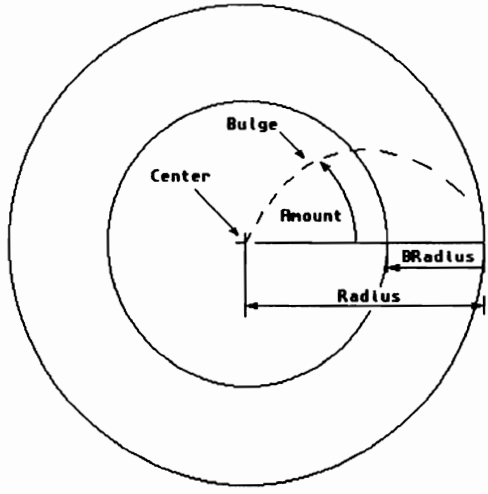


Figure 6.31: Components of the Twirl Circle.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

the radius of this Circle. If you click the left mouse button while the pointer is anywhere but on the crosshair, the radius of the circle will snap to the new location. The actual radius (in pixels) is shown in the panel's **Radius (Px)** input field.

To define how much twirling will be done, you will have to enter a value in the **Amount (Deg)** input field. The current amount of twirl is shown on the Twirl Circle as a "bulge" or arc from the radius line in the direction the twirl will take. This bulge also shows in what direction (clockwise or counter-clockwise) the area defined by the Twirl Circle will be twirled and by how much (in degrees). If a positive **Amount (Deg)** is specified, the arc will bulge upward and the twirl will be in a counter-clockwise direction. If a negative **Amount (Deg)** is specified, the arc will bulge downward, producing a twirl in the clockwise direction. Note that you cannot modify the bulge with the mouse pointer.

The amount (percent) of blurring at the edge of the Twirl Circle can be controlled by the value specified in the **BRadius(0-99)** input field. This is represented as the innermost circle within the Twirl Circle. A value of **0** produces no blurring, whereas a value of **99** produces blurring across the entire radius of the Circle. The position of this inside circle, as well, cannot be modified with the mouse pointer.

The speed (and correspondingly, the quality) of the twirl operation can be controlled with the **Quality** button. Clicking on the button will toggle the type of quality between **Fast** (not the best quality, but faster to complete) and **High Quality** (high quality, but slower to complete).

To abort the display of the current image, select the **Abort** button. Only a portion of the entire image may be shown on screen, but you can still move the Twirl Circle almost anywhere on the image.

To accept the current Twirl Circle values and begin the operation, select the **Accept** button. The progress meter window will appear, which shows an increasing (from left to right) meter bar describing the completeness of the operation. To abort the operation at any time, select the **Abort** button.

If you have finished modifying your image for this session and want to return to the ADPro screen, select the **Quit** button.

6.41 Vertical_Flip

Many applications require the production of mirror images of the original data. Specifically, this is required by screen and heat-transfer printers. Selecting the Vertical_Flip operator will vertically flip the raw image data currently in memory.

Note that performing a horizontal flip followed by a vertical flip is the same as rotating the image 180 degrees. This can be used in conjunction with the orientation (**Port / Land**) button to rotate images through 0, 90, 180, and 270 degrees. Note, as well, that you can also do these same rotations with the Rotate operator.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 7

ARexx Interface

By using the powerful scripting language, ARexx, ADPro becomes an enormously flexible automated image processing system. Nearly every aspect of ADPro can be controlled from ARexx using simple English-like commands. This chapter describes the usage of ADPro from ARexx. For specific ARexx command specifications, please refer to the various sections in Chapter 10.

NOTE You must have ARexx in order to use the facilities described in this chapter and in Chapter 10. Specifically, if ADPro cannot locate the `rexxsyslib.library` file (in the `LIBS:` directory), then it will not respond to any ARexx style requests.

7.1 Addressing ADPro

In order to access ADPro or any of its Loaders, Saver, or Operators from ARexx, you must tell ARexx how to make contact with ADPro. Specifically, you must tell ARexx the name of the message port which ADPro creates for the purpose of ARexx interaction. The name of this port is:

```
ADPro
```

You can make the connection from ARexx to ADPro using the following ARexx command:

```
ADDRESS "ADPro"
```

ARexx will generate an error condition should ADPro not be found.

NOTE The ADPro ARexx port name is case specific. It must be specified exactly as shown above, otherwise interaction will not be possible.

7.2 Getting Results of Commands

After every command directed at ADPro from ARexx is executed, ADPro will fill in the standard ARexx result variable, `RC`, with a return value. If this value is 0, then the previous command executed without error. If the value of `RC` is not zero, then it contains an error severity indicator.

In order to receive additional error indications or to receive non-error status information back from ADPro, you must request additional status information from ARexx using the following command:

```
OPTIONS RESULTS
```

This command must be one of the first executable instructions in your ARexx program.

If an `OPTIONS RESULTS` command has been issued, then ADPro will load additional information into the ARexx variable **ADPRO_RESULT**. **ADPRO_RESULT** can be consulted whenever `RC` is non-zero to gain additional information about the nature of an error.

When `RC` is returned with a 0 value (indicating no error occurred), **ADPRO_RESULT** will be set to command specific information. Note that this means you must check `RC` before depending on the data in **ADPRO_RESULT** since **ADPRO_RESULT** may contain either additional error information or the status information you requested.

The **ADPRO_RESULT** variable is available only to true ARexx programs. If you are sending ARexx style commands to ADPro from a non-ARexx program, then results which normally would be passed back in **ADPRO_RESULT** will instead be passed back in the **rm_Result2** field in the ARexx message structure.

The ARexx command specifications (in Chapter 10) usually list how each command affects the `RC` and **ADPRO_RESULT** variables. If either one or both is missing (or if the variable is labelled "unchanged"), then that particular command doesn't change the unlisted variable's value. If it is labelled "undefined", then the value may or may not have changed. This label means you should not depend on the value of this variable.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

7.3 Launching ARexx Programs

ADPro can launch up to 50 distinct but specially named ARexx programs. These programs must reside in your **REXX:** directory and must be specially named according to the following conventions:

- If any function key (**F1** through **F10**) is pressed, ADPro will attempt to launch an ARexx program named **F1.adpro** through **F0.adpro**. For example, if **F8** is pressed ADPro will attempt to launch **F8.adpro**.
- If any function key (**F1** through **F10**) is pressed along with either **Shift** key, ADPro will attempt to launch an ARexx program named **SF1.adpro** through **SF0.adpro**. For example, if **Shift + F10** is pressed, ADPro will attempt to launch **SF0.adpro**.
- If any function key (**F1** through **F10**) is pressed along with either **Alt** key, ADPro will attempt to launch an ARexx program named **LF1.adpro** through **LF0.adpro**. For example, if **Alt + F3** is pressed, ADPro will attempt to launch **LF3.adpro**.
- If any function key (**F1** through **F10**) is pressed along with either **Amiga** key, ADPro will attempt to launch an ARexx program named **AF1.adpro** through **AF0.adpro**. For example, if **Amiga + F6** is pressed, ADPro will attempt to launch **AF6.adpro**.
- If any function key (**F1** through **F10**) is pressed along with the **Ctrl** (Control) key, ADPro will attempt to launch an ARexx program named **CF1.adpro** through **CF0.adpro**. For example, if **Ctrl + F2** is pressed, ADPro will attempt to launch **CF2.adpro**.

If more than one of the **Ctrl**, **Shift**, **Alt**, or **Amiga** keys are depressed at the same time as a function key, ADPro will *not* attempt to launch an ARexx program.

If one of the key combinations (described above) is depressed for which there is no corresponding ARexx program, you will be told.

We provide an ARexx program attached to the **F10** function key which allows other ARexx programs to be launched by name. This program, called **F0.adpro**, will ask you to specify an arbitrary ARexx program to execute. This allows you to use ARexx programs which do not conform to the naming conventions listed above. The **F0.adpro** program comes on the ADPro distribution disk. You can install it into your **REXX:** directory by running the installation script and choosing the option that installs the sample ARexx programs.

7.4 Understanding the General Rules

Three general rules apply to all ADPro ARexx commands. These are:

1. All commands set **RC** and **ADPRO_RESULT** as indicated above.
2. Most commands which change some internal state in ADPro return the old value in **ADPRO_RESULT** if no error occurred. If a command which requires additional arguments to change an internal state is executed without the required arguments, the old value of the state is returned and is not changed. Therefore, a command which changes some state can also be used to interrogate the present value of the state without changing it.
3. All commands are case-insensitive. However, arguments may be case sensitive.

7.5 Using ARexx Commands

For a complete listing of all ARexx commands available in ADPro, please refer to the various sections in Chapter 10. These ARexx commands are grouped by subject matter—main screen functions, loaders, savers, and operators. Commands which do not belong to any of these groups are listed in Section 10.2.5 in Chapter 10.

Listed below is a complete cross reference of all ARexx commands available in ADPro, along with page references to their descriptions. The commands are listed in alphabetical order.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Command	Description	Page
ABS_SCALE	Scale the image to absolute dimensions.	308
ADPRO_DISPLAY	Display the rendered image.	218
ADPRO_EXIT	Exit the program.	220
ADPRO_TO_BACK	Move the screen behind all others.	221
ADPRO_TO_FRONT	Move the screen in front of all others.	220
ADPRO_UNDISPLAY	Remove the display of the rendered image.	219
BLUE	Set/get the blue content of the rendered image.	222
BRIGHTNESS	Set/get the brightness of the rendered image.	222
CONTRAST	Set/get the contrast of the rendered image.	223
CURRENT_MEM_SIZE	Get the current size of the image buffer.	219
DITHER	Set/get the dither method to use.	232
EXECUTE	Create rendered data from raw data.	218
GAMMA	Set/get the gamma correction for the image.	223
GCR	Set/get Gray Component Replacement value.	283
GETDIR	Ask the user to select a directory.	239
GETFILE	Ask the user to select a file.	237
GETFILES	Ask the user to select a list of files.	237
GETNUMBER	Ask the user to enter a number.	236
GETSTRING	Ask the user to enter a string.	235
GREEN	Set/get the green content of the image.	221
IMAGE_TYPE	Get the type of image data available.	234
INK_CORRECTION	Set/get ink correction value.	284
LAST_LOADED_IMAGE	Get the filename of the last loaded image.	233
LAST_SAVED_IMAGE	Get the filename of the last saved image.	233
LFORMAT	Set/get the loader format.	243
LOAD	Load an image into ADPro.	243
LOADER	Execute a loader module.	245
MAGENTA_CORRECTION	Set/get magenta correction value.	284
OBTAIN_ADPRO	Obtain exclusive lock on ADPro.	241
OFORMAT	Set/get the operator module.	286
OKAY1	Display a message with one button.	240
OKAY2	Display a message with two buttons.	240
OPERATE	Perform the currently selected operator.	287
OPERATOR	Execute an operator module.	287
ORIENTATION	Set/get the load orientation.	245
PAUSE	Cause a delay in the execution of the script.	235
PCONTRAST	Define whether colors 0 and 1 should contrast.	226
PCT_SCALE	Scale the image by percentages.	308
PGETWB	Load the first four Workbench colors.	228
PIXEL_ASPECT	Get the image's pixel aspect.	240

Command	Description	Page
PIXEL_RESOLUTION	Get the image's pixel resolution.	241
PLOAD	Load a palette file.	228
POFFSET	Specify a register as offset color zero.	226
PPOKE	Set/get the value in a color register.	227
PSAVE	Save the contents of the palette to a file.	229
PSORT	Set/get the sort ordering of the palette.	227
PSTATUS	Set/get the palette lock mode.	224
PTOTAL	Set/get the total number of colors.	225
PUSED	Set/get the number of colors to actually use.	225
PWIDTH	Set/get the accuracy of the palette.	224
RED	Set/get the red content of the image.	221
RELEASE_ADPRO	Release exclusive lock on ADPro.	241
RENDER_TYPE	Set/get the rendering mode.	217
SAVE	Save the image currently in ADPro.	261
SAVER	Execute a saver module.	262
SCREEN_TYPE	Set/get the type of screen to use.	230
SEPARATE	Perform a color separation.	285
SEPARATION_DEPTH	Set/get color separation depth.	285
SEPARATION_MAP	Set/get color map type.	283
SEPARATION_TYPE	Set/get color separation type.	285
SERIAL_NUMBER	Get the program's serial number.	219
SFORMAT	Set/get the saver format.	261
UCR	Set/get Under Color Removal setting.	283
USE_SEPARATION_DEFAULTS	Set UCR/GCR/ink correction to default values.	285
VERSION	Get the program's version number	219
XSIZE	Get the width of the image.	234
YELLOW_CORRECTION	Set/get yellow correction value.	284
YSIZE	Get the height of the image.	234

7.6 Working With ParseArgs

Most of the ARexx commands return information in **ADPRO_RESULT**, especially when an error has occurred. The release of ADPro to the public marks the first major distribution of modules which use a slightly different form of ARexx command invocation and error detection. Known as ParseArgs, this technique of interfacing with a program (or module) through ARexx should make the task of writing ARexx programs more flexible.

Not all of the modules (loaders, savers, operators) use ParseArgs to handle command parsing. Most of the time, you will be able to tell which method (old or ParseArgs) is being used just by the freedom with which you can specify module-specific options or arguments. In the old style, these arguments usually had to be given in a defined order. In the new, ParseArgs, style, these same

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

arguments can be listed in any order (within specific constraints).

For example, the ARexx interface for the Roll operator (an old-style interface) is:

OPERATOR "ROLL" *direction pixels* or
OPERATOR "ROLL" *direction pixels* **"WRAP"**

Notice how the *direction* value must come immediately after the **OPERATOR "ROLL"** sequence and before the *pixels* value. Under a ParseArgs-style interface, that same module could be controlled with the following syntax:

OPERATOR "ROLL" *arguments*

where *arguments* can be zero or more of the following (listed in any order):

DIRECTION *direction*

PIXELS *pixels*

WRAP

If one of the above *arguments* were missing, then ParseArgs would just use the last value used for that particular *argument*. Also, being able to list these arguments in any order frees you from having to remember which argument should come before another.

Another advantage that ParseArgs modules have over their old-style counterparts is the improved error handling. Say, for instance, you forgot to include the required **PIXELS** argument in your ARexx command. ParseArgs would sense this, give you an error code in **RC**, and give a descriptive message in **ADPRO_RESULT**, such as, **"No arg found in same class as: PIXELS"**. From this information, you would be able to determine which arguments need to be specified. If more than one required argument is missing, however, only one of the missing required arguments will be described in **ADPRO_RESULT**.

In addition to checking for required arguments, ParseArgs also confirms that given values are within the valid range of values (as specified by the module itself). For example, say you typed in **PIXELS 500** but the valid range was only **0** to **250**, ParseArgs would (again) tell you what is required in **ADPRO_RESULT**.

7.7 Using the Supplied ARexx Programs

f0.adpro
: Executing this program brings up the file requester in the **REXX:** directory allowing you to pick (by name) another ARexx program

to execute. We find this program so useful, we bind it to your **F10** function key (by calling it **F0.adpro**).

NOTE You may have to modify this program so that it points correctly to your RX program. See the program for more information.

getfiles_example.adpro

This program is an example of using the **GETFILES** ARexx command. **GETFILES** is like **GETFILE** in that it brings up the file requester, but **GETFILES** allows the user to select multiple files at one time.

This ARexx program will ask you to select multiple files and then steps through your selections.

last_loaded.adpro

This program makes use of the **LAST_LOADED_IMAGE** command to tell you the name of the file you last loaded.

locate-adpro.adpro

This program is an example of how you might start ADPro from ARexx if you discovered it wasn't running at the time your own ARexx program began executing.

scaletoaspect.adpro

This program is an answer to those many people who wanted an automatic way to scale to a specific pixel aspect. As long as the pixel aspect of your image data is set correctly in the first place, this ARexx program figures out how to scale the image to the aspect you want.

For example, suppose you scanned an image (from a 1:1 pixel aspect scanner) and you want to scale the image for display in a hi-res interlaced screen. Simply tell this program you want a 22 to 26 image (when it asks you) and it handles the rest.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 8

FRED

FRED is a visually oriented list manager. The lists it manages, called sequences, are comprised of image names and other information. Individual images (or frames) in a sequence are represented on-screen with accurately rendered icons.

FRED is an independently executable program which can be run in conjunction with ADPro.

FRED can instruct ADPro to process each frame in a sequence, providing an easy method of batch processing. FRED also provides an extensible ability to call special purpose drivers which can cause ADPro to generate animation effects automatically. You can even preview animations within FRED, before (for example) committing them to a single frame recorder.

FRED is a powerful tool for the 24 bit single animator. It also benefits those who would like to batch process unrelated pictures but feel they are not up to the task of writing the appropriate ARexx program.

NOTE FRED requires AmigaOS 2.04 or later of Commodore's Amiga operating system. FRED also requires that ADPro be running at the same time for it to run.

To start FRED, double-click on its icon. A separate control screen will appear.

Before we describe the features of FRED, you should know how to exit the program. The **Quit** menu item will do this for you.

To find out which version of FRED you are running, select the **About...** menu item. A window will appear with this information, as well as a legal notice.

Two sets of menus are accessible from within FRED. When the backdrop screen is active, a menu set including **Project**, **Options**, and **AnimOps** items can be accessed. When a sequence window is active, a menu set including **Project**, **Edit**, and **Miscellaneous** items can be accessed.

In both sets, the **Project** menu is the same.

8.1 Sequences, Cells, and Frames

Sequences, as mentioned earlier, are lists of images that will be processed by FRED. These sequences can be assembled, loaded, saved, processed, and animated.

Creating a new sequence is as easy as selecting the **New...** menu item (from the **Project** menu). An empty window, called the sequence window, will appear. Its title bar displays the name of the sequence (initially defined as `UNNAMED`*xxx*`.seq`, where *xxx* is a number from 000 to 999), as well as if the sequence was modified at all (`[*]` for a modified sequence, `[ ]` if not modified).

These sequences are made up of cells (single images). These images can be similar to one another (such as cells of an animation) or a collection of varied images. What they share in common is the need to be processed.

Images can be inserted into a sequence as follows. Click on the sequence window to add to; the window should now be active. Select either the **Insert Image(s)...** or **Insert Range...** menu item. **Insert Image(s)...** lets you choose one or more images (from the same directory) to insert into the currently active sequence. **Insert Range...** allows you to insert a group of images that have numeric file extensions. For example, you can select to load `mypic.0004` (the first image in the range) through `mypic.0020` (the last image) and FRED will automatically search for files with a base name of `mypic` and extensions of `.0004`, `.0005`, all the way up to `.0020`. Note that the selected images will be in inserted *after* the currently selected cell, which has its cell information box depressed and its text displayed in white.

If you want to save the defined cells of the sequence, select either the **Save** or **Save As...** menu item. **Save** will store the cells under its current sequence name (as shown in the window's title bar), if it has one. **Save As...** will let you save the sequence under a new name.

If no window is currently active (i.e., the backdrop screen is active), a list of currently open sequences will appear. If you select a sequence name and then press the **Accept** button, that sequence will be saved.

To open a previously saved sequence, select the **Open...** menu item and select the sequence to load from the file requester that will appear.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

You can have many sequence windows open on the FRED screen as graphics memory will allow. These windows can be hidden from view (also termed "iconified") by clicking on a window's zoom gadget.

To close a sequence window, select the window and choose the **closE...** menu item.

To make a window reappear or to activate another open window, select the **Switch To...** menu item. A list of currently open sequences will be displayed. If an 'I' appears before a sequence's name, the window corresponding to that sequence is iconified. Select the sequence to switch to and press the **Accept** button. You can also double-click on the sequence name to bypass the **Accept** button altogether.

8.1.1 Cells vs. Frames

An important distinction in terminology must be made at this point before you continue reading the rest of the FRED documentation.

So far, we have termed the individual images of a sequence as "cells". What hasn't been discussed (but what will be later) is that each cell does not have to be a single instance of an image—it can represent the same image as if it had been entered as successive cells, *without having to insert them.* This is accomplished through the use of "frames."

For example, say that you want a particular cell to represent five instances of an image. Instead of adding five cells to the sequence, you can add just one and specify a length (in the **Length** input field of the Image Info control panel) of 5. This ability comes in handy when animating the selected cells and batch processing the images.

8.2 Selection and Arrangement of Cells

Any cell in a sequence can either be selected or unselected. Any number of cells can be in the selected range. The currently selected (active) cell has a depressed information box with its text shown in white, whereas the cells in the current range have the background of their text (in their information boxes) shown in the highlight color.

To select a cell as the current one, simple click on it. Its information box will be depressed and its text shown in white. If you want a particular cell to be placed in the selected range, hold down the `Ctrl` key when clicking on the cell. The background of the text will become green, but the text color will stay the same (black). To remove a cell from the range, do the same operation.

You can also select a specific range of cells by clicking on the first cell in the range, selecting the **Select Range** menu item, and clicking on the last cell in the range. All of the cells between, and including, these two will be highlighted.

NOTE When FRED works on the selected range, it will use the highlighted cells *as well as the current cell, even if the current cell's text is not highlighted.* This is an important point to remember.

You can also select all of the cells in a sequence by issuing the **Select All** menu item. Deselecting all of the cells is just as easy with the **Deselect All** menu item.

Cells do not have to stay in the same arrangement that they were added or inserted into the sequence—you can use the Clipboard to cut and copy cells from one part of the sequence and paste them in other parts. Note that only the information describing the cell, and **not** the image itself, will be posted to or retrieved from the Clipboard.

To remove a set of cells, select the cells and choose the **Cut** menu item. If you just want to copy the cells, then select **Copy** instead.

Pasting cells is done by selecting the cell that will preceed the pasted information and selecting the **Paste** menu item. If no cell is currently selected, the pasted cells will be appended to the end of the sequence.

You can also paste the cells in reverse order that they were placed into the Clipboard by selecting the **Reverse Paste** menu item.

NOTE All cells that you cut or copy then later paste (using either paste command) keep their images' filenames. This is very important if you are going to process these images and overwrite the original version with the processed ones.

8.3 Cell Information

The **Set Text Options...** menu item will bring up the Set Text Options control panel (shown in Figure 8.1), from which you can change the text which appears below each stamp in a sequence as well as what text will be displayed during animation previews.

Both cycle gadgets (**Edit Label** and **Anim Label**) will let you select one of the following:

Root Name This is the root filename of the image, as it can be found on disk. This filename does not include any path information.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

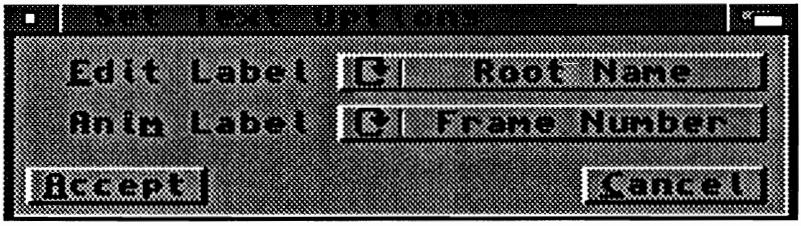


Figure 8.1: The Set Text Options control panel.

Frame Number A cell's frame number is a count of the number of "recordable units" which come before this cell. The first frame (first cell) in a sequence is frame zero. Each cell's length determines how many recordable units each frame occupies.

For example, if a sequence contained just two cells each having a length of 1, then the frame number fields will show "00000000" and "00000001". If the first cell had a length of 60, then the cell's first frame number would still show "00000000" but the second frame number would show "00000060".

Time This shows the exact time at which the image will be displayed in the animation. The computation of time is based upon the current animation frame rate.

Time is expressed as minutes, seconds, and frames (or mm:ss:ff). A frame is a fraction of a second, whose duration is determined by the currently selected frame rate.

Length This shows the number of recordable units in the animation that the current stamp will be displayed.

Size This is the width and height (in pixels) of the image. This information is available only if a stamp has been created for this image. "Stamp" is another word for the icon which FRED uses to represent each picture.

The **Change Duration...** menu item displays the Change Duration control panel (shown in Figure 8.2) from which you can change the number of recordable units the currently selected cells will occupy in the animation. All cells have a default time of 1.

The amount of time a given cell's frames will actually be displayed depends upon the currently specified frame rate. For example, if the current playback

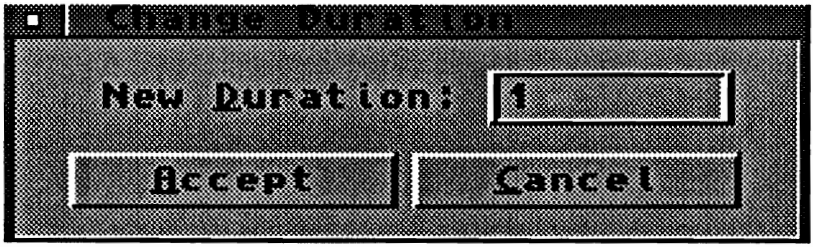


Figure 8.2: The Change Duration control panel.

Figure 8.3: The Image Info control panel.

speed was 30 frames per second, an image having a time of 15 will be displayed for half a second when the animation is played.

This Change Duration value can also be entered in the Image Info control panel (shown in Figure 8.3). To bring up this panel, either double-click on the cell itself, or single-click on the cell and either select the **Image Info...** menu item or press the **Return** key. In the Image Info control panel, you will see the filename of the cell, its currently defined length, the **Time In:** and **Time Out:** values, the **Frame In:** and **Frame Out:** values, and its **Width:** and **Height:**.

The **Time In:** and **Time Out:** values specify the display time (in seconds) for this cell. It takes into account the current **Animate Speed** (see Section 8.4) you have specified. For example, say that you have specified a speed of **15 Frames Per Second** and that the first three cells are of length 10, 5, and 20, respectively. The time in/out (frame in/out) values for these frames would be:

Frame		Time	
In	Out	In	Out
00000000	00000009	00:00:00	00:00:09
00000010	00000014	00:00:10	00:00:14
00000015	00000034	00:01:00	00:02:04

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The **Frame In:** and **Frame Out:** values specify the first and last frame number described by this cell.

8.4 Animated Stamps

FRED allows you to preview your sequence of images in small, stamp-size images. You can specify the type of animated preview to produce, as well as the FPS (frames per second).

To create stamps for the currently selected cells, select the **Make Stamp** menu item. ADPro must be currently running for this operation to work. Each stamp will be placed in the same directory as the image it is making a stamp for, using a file extension of "`.stp`" to denote that it is a stamp.

If you are running on a non-AGA machine (such as the A3000), the stamps will be rendered in 4 bit-planes (or 16 colors). If, however, you are running on an AGA-equipped machine, then any generated stamps will be rendered in 8 bit-planes (or 256 colors).

Once all of the stamps have been created, you can animate them by selecting various settings and operations in the **Miscellaneous** menu.

The **Miscellaneous** Menu contains commands which allow you to set the frame rate (which affects how time is computed elsewhere), preview animations, create the stamps viewed in FRED, and cause generalized batch processing to take place. This menu is accessible only when a sequence window is selected.

You can choose to animate the entire sequence or the currently selected range by choosing either **Entire Sequence** or **Selected Range**.

The speed of the animation playback can be defined by choosing the desired frame rate from the **Animate Speed** menu item's subitems. Thirty FPS (frames per second) is the standard for NTSC video playback. Twelve FPS is useful for previewing animations which will be displayed on Commodore's CDTV. If FRED is running on a PAL machine, the options will be 25, 15, 12.5, and 10 FPS.

The setting you choose here affects how time is calculated elsewhere in FRED. If you set 30 FPS, for example, FRED will increment the seconds counter (in the time code display) once each 30 frames. Setting the frame rate to 10 causes the seconds counter to increment every 10 frames.

To actually animate the frames using the current animation settings, select one of the three types of motion: **Play Once, Continuous**, and **Ping Pong**. **Play Once** will play the frames only once. **Continuous** will play them over and over again, continuously. The last frame is followed by the first frame.

Ping Pong is similar to **Continuous**, except that the frame order reverses upon displaying the last frame.

A small window will appear, showing the animated frames. If you find that the frames are flickering, move the window to the bottom of the screen. This should reduce this problem. To stop the animation, click on its close gadget.

8.5 The ADPro-FRED Link

NOTE To process images with FRED and ADPro, you must be knowledgeable in writing simple ARexx scripts. Consult the ADPro manual (and even Chapter 7 in this manual) for more information on what needs to be included in a FRED ARexx script.

Your ARexx programs needs not concern themselves with the details of loading the image to be worked upon, nor do they need to know how to pick the next image to be processed. These tasks are performed by FRED before calling the first ARexx program in the chain you specify.

Arguments are passed to each ARexx program which indicate a frame's attributes. Use the ARexx `ARG` command as follows to fetch these values:

```
ARG FrameNum FrameFName Length LoadFlag FirstCall
```

`FrameNum` will contain the current frame number (starting from 0). `FrameFName` will contain the current frame's filename. `Length` will contain the length (number of frames) of the current cell. `LoadFlag` will contain either 1 if the current image was loaded or 0 if the wasn't. This lets you determine the setting of the **Load** cycle gadget in the Invoke ADPro control panel. `FirstCall` will be equal to 1 if this is the first frame being processed. You can use these variables in your ARexx script to display meaningful error messages and customize filenames used with the `SAVE` command. Consult the sample ARexx scripts included with ADPro for examples on how these values can be used.

FRED allows you to process either all of the images in a sequence or a select few. Note that as with the other commands that reference the currently selected cells, the currently selected cell (shown with its text in white) is part of the selected group, even if the background of its text is not highlighted (green).

Choose the images you want to process and select the **Invoke ADPro...** menu item. The FRED Script List control panel (shown in Figure 8.4) will appear with the currently defined list of ARexx scripts. Notice that you can create a chain (list) of scripts (with the **Add...**, **Insert...**, and **Delete** buttons) that will be executed for each selected image in the current sequence. This

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

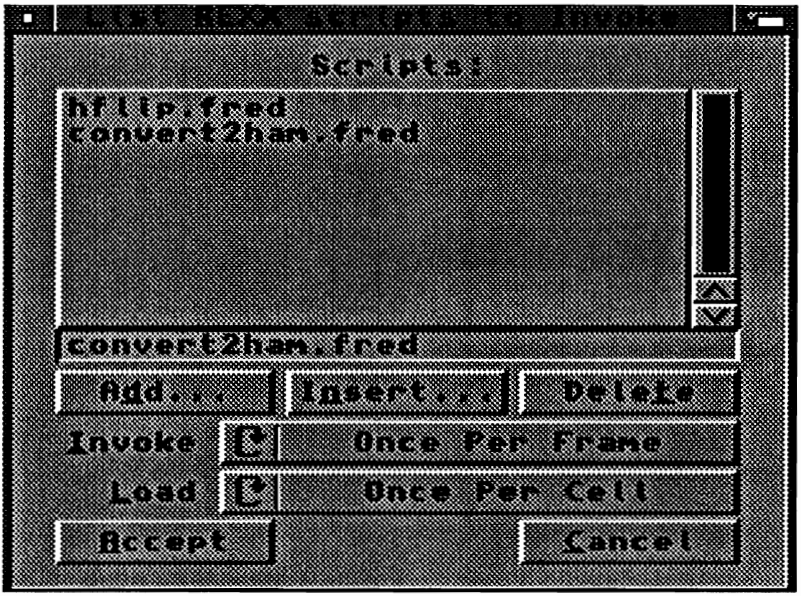


Figure 8.4: The Invoke ADPro control panel.

allows you to assemble a complicated sequence of operations with smaller, more simpler, and more generalized scripts which you can reuse.

The buttons and cycle gadgets in this control panel are described below:

Add... A file requester will appear. Use it to enter the name of the ARexx script you would like to add. The selected ARexx program is added to the end of the chain.

Insert... This command allows you to add an ARexx script into the middle of the chain. First select the place in the chain where you would like to insert the script, then click on the **Insert...** button. A file requester will then appear in which you can enter the script name you would like to insert.

Delete This allows you to delete a script from the chain. First select the name of the script to be deleted, then click on the **Delete** button to remove the script from the chain.

Invoke This cycles between **Once Per Frame** and **Once Per Cell**, indicating when to invoke the list of scripts.

Load This cycles between **Every Frame** and **Once Per Cell**, indicating when to load each image.

Once you are satisfied with the list of scripts and are ready to process the selected images, press the **Accept** button. A meter window will appear, showing the progress of the operations.

If you want to return to the FRED control screen without performing the operations, select **Cancel**.

8.5.1 Pre and Post Processing

In addition to the processing that will be performed on each selected frame in the sequence, you can also specify pre-processing and post-processing scripts. These scripts, along with the usage of ARexx's Clip List entries (attribute-value pairs), allow you to customize your Invoke ADPro scripts based on user input.

Pre-process scripts will be performed before any image data is loaded. Typically, you would use pre-scripts to ask the user to select certain options or set values used by the Invoke ADPro scripts.

Post-process scripts will be performed after all frames are processed. Typically, you would use these scripts to clean up the working environment, such as deleting temporary files or resetting or restoring Clip List entries.

These pre and post scripts are not specified directly by name, but are selected as a result of placing a particular script in the Invoke ADPro list of scripts. For example, if the "**SaveAsANIM.fred**" script (included with this package) was entered into the list, when you instruct FRED to execute the scripts it will try to find a script with a similar name, but with "**.pre**" appended to it (in this case, "**SaveAsANIM.fred**"), to execute before any images are processed. After doing this pre-processing, FRED will execute the "**SaveAsANIM.fred**" script for each selected frame in the sequence. Once done, it will try to find and execute a script with a similar name, but this time with "**.post**" appended to it.

NOTE When you select scripts to insert into the Invoke ADPro list, you are implying that the corresponding pre and post scripts should be executed as well. So, if you want a script to handle pre-processing for a given script, be sure to append ".pre" to the name. For post-processing, be sure that you use ".post".

The order with which FRED will invoke the scripts is as follows:

1. All pre-process scripts for all defined Invoke ADPro scripts;
2. For each selected frame, execute the scripts listed in the Invoke ADPro list;

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

3. All post-process scripts for all defined Invoke ADPro scripts.

Included with the package are a set of "canned" scripts which you can use on your images. We hope that with these scripts you will find many more uses for FRED.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 9

AnimOps

The **AnimOps** menu is accessible only when the backdrop is selected and contains the name of each program found in the AnimOps directory (usually `ADP_FRED:AnimOps`).

AnimOps are programs which read sequence files and generate ARexx commands automatically controlling ADPro. Any programmer may create an AnimOp. Contact ASDG for more details.

The AnimOps supplied by ASDG with FRED are described in their own sections.

In this chapter, we will discuss the AnimOps supplied by ASDG with FRED. While all AnimOps should have certain elements in common (that is, if their authors have followed ASDG's design guide), details concerning AnimOps supplied by third parties will not be contained here.

9.1 Compositor

The Compositor allows you to easily create combinations of fades, wipes, and composites between two or more sequences of images using FRED. Users of previous version of FRED may be familiar with the Compositor and Alpha_Compositor AnimOps that were included with the first versions of the program. This Compositor supersedes both of them, combining them into one AnimOp and including more options and controls.

The Compositor allows you to work on compositing "projects," which can be created, stored, modified, and retrieved. Select the **Compositor** menu item

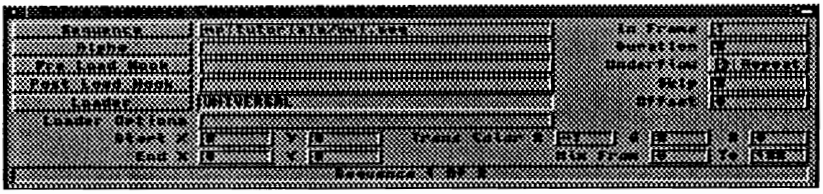


Figure 9.1: The Compositor AnimOp's Components control panel.

from the **AnimOps** menu. A thin window bar will appear atop the FRED screen. All of the main controls are available from the AnimOp's menu bar.

9.1.1 Projects

To create a new compositing project, select the **New...** menu item. The small AnimOp window bar will expand to the Results control panel. This panel will be discussed in Section 9.1.3.

Projects can be created, stored on disk, opened, closed, and edited. All of the controls for projects are available in the **Project** menu.

To define the compositing of sequences, you must go through two steps:

1. specify the sequences to be composited
2. specify the resulting sequence to create

The first step will be described in Section 9.1.2, while the second will be in Section 9.1.3.

9.1.2 Components

To display the Components control panel (as shown in Figure 9.1), select the **Define Components** menu subitem (from the **Mode** menu item). It is in this panel where you specify how one or more sequences will be composited together.

You have complete control over when, where, and by how much each sequence's frames will mix with the frames of other sequences. Let's take a look at the contents of this panel.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The first thing you should spot is the **Sequence Indicator** text field at the bottom of the panel. It is here where you can identify one sequence from another. You can flip from sequence definition to definition by using the **Next** and **Previous** menu items (in the **Edit** menu). This field will display which sequence (of the total number of sequences) you are currently editing.

Sequences are numbered starting from 1. The sequence defined as Sequence 1 can be thought of as being at the bottom (position-wise) of the sequences that will be defined. The other menu items in the **Edit** menu allow you to add, delete, copy, paste, and erase sequences from this "stack" of sequences.

The sequence file is described in the **Sequence** input field. You can either enter a new filename into the field or press the button to select one from the file requester. Only files ending in `.seq` will be visible. When saving sequence files out of FRED or any other program which outputs sequence files, you must ensure that this filename extension exists for FRED and these other programs to recognize it as a valid sequence file.

You can specify the transparency of the images in the sequence with another sequence of grayscale images (of the same dimension as the images in the sequence to composite) by entering the sequence's filename in the **Alpha** input field. This alpha channel sequence file should be of the same length as the primary sequence itself.

The **Pre Load Hook** and **Post Load Hook** input fields can contain the filenames of ARexx scripts which will be executed before and after loading each frame of the sequence. This feature gives you a lot of flexibility in manipulating the images in your sequence, without having to modify the originals—you can use the same master sequence with different hooks to produce a potentially drastic difference set of images.

The **Loader** and **Loader Options** text and input fields allow you to specify the type of images that are contained within the current sequence, and any loader-specific arguments which might be required. By default, the UNIVERSAL loader is used.

The **Start X** and **Y** values control the initial position of the upper left hand corner of the images in this sequence, relative to the currently defined "backdrop" image. The **End X** and **Y** values specify the final position of this same corner for the last frame to be processed in this sequence.

The "backdrop" image for a given frame number will not always be the image defined in the bottommost sequence (Sequence 1). Depending upon the **In Frame** number (described later in this section), a "higher" sequence might actually constitute the backdrop.

The transparent color in the current sequence's images can be defined with the **Trans Color R**, **G**, and **B** values. If **R** is set to -1, then there will be no

transparent color (the image will be opaque). If each of these values are set to a number between 0 and 255, then any pixel in the image that is this color will be transparent, and that the next lowest opaque pixel (from lower sequences) will "show through." Note that if a current frame is the backdrop image, then these transparency values are ignored.

The amount of mixing that will be done, from the first frame to process to the last, is defined in the **Mix From** and **To** input fields. These values can range between 0 and 100.

The actual frames (in this sequence) that will be composited is controlled by the values in the input fields and cycle gadget located at the right of the panel.

In Frame describes at which frame number (sequences are described as starting with a frame number of 1) in the resulting sequence this current sequence's frames will start to be composited. The first frame (in this sequence) to process is specified in the **Offset** input field. The number of frames to process is specified in the **Duration** field. If you want to process every other frame, or every third or fourth frame, then you can change the value in the **Skip** input field to tell the AnimOp how many frames to skip after finishing with a particular frame.

If the resulting sequence "fills up with images" (the total number of frames requested has already been reached) but you still have frames in the current sequence to process, then those frames will not affect the final sequence at all. If, however, you use up all of the selected frames in the sequence before the resulting sequence finishes, then the setting of the **Underflow** cycle gadget will define what to do with the current sequence.

If set to **Repeat**, then the last frame that was processed will be used for the rest of the resulting sequence. If set to **Loop**, then the current sequence will act as if the sequence was never-ending (i.e., the sequence would start at the beginning again). If set to **Stop**, then no more frames from this sequence will take part in the compositing.

9.1.3 Results

Once you have defined the sequences that will be composited together, you would then define the resulting sequence that will be generated. Switch to the Results mode by selecting the **Define Results** menu subitem. The Compositor's Results control panel (as shown in Figure 9.2) will appear.

The name of the resulting sequence must be defined in the **Output Sequence** input field. The composited frames will be saved (appended by the sequence's current frame number) with a base (root) name, as defined in the **Image Root Name** input field. If you want the frames to be saved as

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Figure 9.2: The Compositor AnimOp's Results control panel.

Work:Images/vacation.00001, **Work:Images/vacation.00002**, and so forth, then enter **Work:Images/vacation** as the root name.

If you require special processing of the image before or after it is saved to disk, then specify the ARexx scripts in the **Pre Save Hook** and **Post Save Hook** fields, respectively.

The format in which these images will be saved must be specified in the **Saver** text field. Any saver-specific options that need to be included should be entered into the **Saver Options** input field.

Lastly, the total number of frames in the resulting sequence is defined by the value in the **Total Frames** input field.

Once you are satisfied with the project, you may want to save it before actually initiating the compositing. To begin the process, select the **Accept** menu item (from the **Project** menu). To exit this AnimOp, select **Quit**.

9.2 TimeStretch

The TimeStretch AnimOp (also referred to as the "time stretcher") allows you to increase or decrease the number of frames in a given sequence by merging or interpolating image data. For example, suppose you rendered an animation at 10 frames per second. Playing this animation back at 30 FPS either results in triple speed (if you display each frame for $\frac{1}{30}$ of a second) or jerky motion (if you display each frame for $\frac{3}{30}$ of a second).

TimeStretch can create additional frames containing in-between multiple exposures easing from one original frame to the next. If the results are played back at 30 FPS, the animation will look smoother than the original (though, of course, not as good as if you had actually rendered it at 30 FPS in the first place).

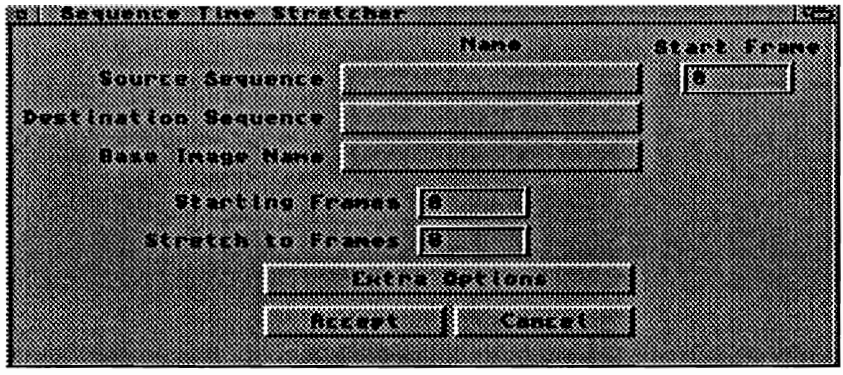


Figure 9.3: The TimeStretch AnimOp's main control panel.

On the other side of the coin, TimeStretch can shorten time. Suppose you have a 30 FPS animation but you know your display device cannot possibly display 30 FPS (but can achieve 15 FPS). You could correct for this by simply leaving out every other frame but this will make you animation look choppy.

You can tell TimeStretch to change 30 FPS into 15 FPS. It will produce a sequence containing the information of two 30 FPS frames in every 15 FPS frame. The results look much more fluid than simply dropping frames.

9.2.1 Main Control Panel

The main control panel (shown in Figure 9.3) allows you to select the sequence to stretch, as well as define to how many frames it should be stretched.

Source Sequence This is the sequence which you would like to time stretch. To select a sequence, select the **Source Sequence** button and a file requester will appear for you to select the sequence name.

Start Frame This is the frame offset at which TimeStretch will begin. This allows you to stretch a portion of a sequence, affecting only the frames including and after the offset number. The offset must be less than the total number of frames in the sequence.

Destination Sequence This is resulting sequence produced by TimeStretch. To specify a filename, select the **Destination Sequence** button and a file requester will appear. If you are recording directly to a single frame device, you can leave this field blank.

Base Image Name This is the base name of the images which are created in the destination sequence. This allows you to specify a name for the images that were created by the compositor. If the base name was `pic`,

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

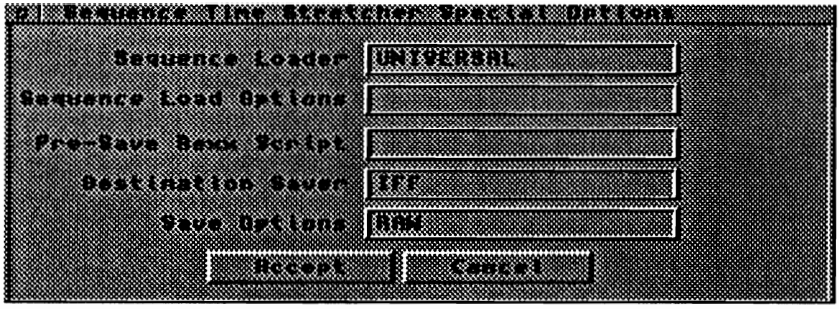


Figure 9.4: The TimeStretch AnimOp's Extra Options control panel.

all of the images would have the name `pic` with the frame number as a suffix. For example, the image in frame number 30 of the sequence created would be named `pic.00030`.

Starting Frames This is the number of frames in the source sequence which you would like to stretch. This number may not exceed the length of the source sequence. A beginning frame number other than zero may be specified by entering a number in the **Start Frame** input field. This will allow you to take a portion of a sequence instead of the entire sequence.

Stretch To Frames This is the number of frames you would like TimeStretch to create from the source sequence. If you specify more frames than are in the original sequence, the new sequence will be become longer. If the sequence has more frames than the number of frames specified, the sequence will be compressed.

9.2.2 Extra Options Control Panel

This control panel (shown in Figure 9.4) allows you to customize the ARexx commands that FRED will send to ADPro. For instance, if you know the background sequence contains only images in the IFF format, then you might specify IFF as the background loader rather than UNIVERSAL, an ADPro-only loader, which will speed things up a little.

Background Loader This specifies the loader that ADPro will use when loading the images for the background sequence. Any valid loader name in ADPro may be used.

Background Load Options This enables you to specify the loader options ADPro will use. To specify an option refer to the ARexx section of the ADPro manual. The loader options are the same as the ARexx options for each of the different loaders.

Foreground Loader This specifies the loader that ADPro will use when loading the images for the foreground sequence. Note that the foreground and background loaders do not have to be the same.

Foreground Load Options This enables you to specify the loader options ADPro will use. To specify an option refer to the ARexx section of the ADPro manual. The loader options are the same as the ARexx options for each of the different loaders.

Pre-Save ARexx Script You can specify the name of an ARexx program here which will be called immediately after the compositing functions are completed but prior to saving the results. You can tie in single frame recorders here, for example, by causing the image to be saved to a display device and then issuing the ARexx command to record.

Destination Saver This is the file format you would like to use for saving composited images. You may specify any of ADPro's savers. Blank out this string if you don't want the AnimOp to directly cause a save to take place.

Save Options This allows you to specify the options that the saver will use. These arguments may be found in the ARexx section of the ADPro manual or reference section of the ADPro manual.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Chapter 10

ADPro Reference

This chapter describes all the control panels, menu items, and ARexx commands available in ADPro.

10.1 Starting ADPro

The ADPro program can be invoked by either double-clicking on its Workbench icon or typing its name at a Shell prompt. You can also customize various parts of the program by specifying some Tool Types or using a few command line arguments.

Tool Type Options

You can specify Tool Type options (using ADPro's icon) to control ADPro. These include:

BEHIND=*n* or **BH=***n*

If *n* is non-zero, the ADPro main screen will be opened behind all other screens open at the time.

MAXMEM=*n* or **MM=***n*

If present, ADPro will be limited to using *n* bytes for its image buffer size. Do not include comma separators in the *n* value. If there is less than *n* bytes free, the buffer size will be equal to the available number, not *n*.

NOENHANCED=*n* or **NE=***n*

If *n* is non-zero, the enhanced palette option will be disabled.

DEFAULTFILE=*filename* or **DF=***filename*

If present, the defaults will be loaded from the *filename* specified.

NOLOADDEFAULTS=*n* or **NL=***n*

If *n* is non-zero, the **ADProDefaults** file will not be loaded when ADPro is started.

NOSAVEDEFAULTS=*n* or **NS=***n*

If *n* is non-zero, the **ADProDefaults** file will not be saved when ADPro exists.

SHALLOW If defined, the ADPro screen Visual operators' screens (consult the section listing these modules) will only open in 3 bit-planes, even though they can open using a deeper 8 bit-plane screen.

ADPROSHALLOW If defined, the ADPro screen will be limited to 3 bit-planes, even though it can render in 8 bit-planes.

VISUALSHALLOW If defined, the Visual operators' control screens will be limited to 3 bit-planes, even though they can render in 8 bit-planes.

SUPER72 If defined, the ADPro screen will open in Low Res Non-Interlace Super72 mode (400 x 300) and the visual operators will open in High Res Interlace mode (800 x 600). If your Workbench screen is already in this mode, then you do not need to specify this Tool Type to get Super72 mode.

ADPROSUPER72 If defined, only the ADPro screen will open in Low Res Non-Interlace Super72 mode (400 x 300).

VISUALSUPER72 If defined, only the Visual operators' control screens will open in High Res Interlace Super72 mode (800 x 600).

Command Line (Shell) Arguments

Alternatively, you can use one of more of the Command Line (Shell) options :

BEHIND

If present, the ADPro screen will be opened behind all other screens.

MAXMEM=*n* or **MM=***n*

If present, ADPro will be limited to using *n* bytes for its image buffer size. Do not include comma separators in the *n* value. If there is less than *n* bytes free, the buffer size will be equal to the available number, not *n*.

NOENHANCED or **NE**

If present, the enhanced palette option will be disabled.

DEFAULTFILE=*filename*

If present, the defaults will be loaded from the *filename* specified.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

NOLOADDEFAULTS

If present, the **ADProDefaults** file will not be loaded when ADPro is started.

NOSAVEDEFAULTS=*n*

If present, the **ADProDefaults** file will not be saved when ADPro exits.

SHALLOW If defined, the ADPro screen Visual operators' screens (consult the section listing these modules) will only open in 3 bit-planes, even though they can open using a deeper 8 bit-plane screen.

ADPROSHALLOW If defined, the ADPro screen will be limited to 3 bit-planes, even though it can render in 8 bit-planes.

VISUALSHALLOW If defined, the Visual operators' control screens will be limited to 3 bit-planes, even though they can render in 8 bit-planes.

SUPER72 If defined, the ADPro screen will open in Low Res Non-Interlace Super72 mode (400 x 300) and the visual operators will open in High Res Interlace mode (800 x 600). If your Workbench screen is already in this mode, then you do not need to specify this Tool Type to get Super72 mode.

ADPROSUPER72 If defined, only the ADPro screen will open in Low Res Non-Interlace Super72 mode (400 x 300).

VISUALSUPER72 If defined, only the Visual operators' control screens will open in High Res Interlace Super72 mode (800 x 600).

10.2 ADPro Control Screen

ADPro's main (control) screen contains all of the controls for loading and saving images and selecting and executing operators. The screen and colors controls, as well as various other miscellaneous controls are available from the main screen.

10.2.1 Commands

ARexx Interface

Control over the Load and Save Formats, the Operator Type, as well as the Load, Save, and Execute Op operations will be discussed in their own sections.

Render Type

Syntax: **RENDER_TYPE** (1)
RENDER_TYPE *rvalue* (2)

The first form of the command returns the current rendering type. This result will be returned in **ADPRO_RESULT** and may be one of the following: **2, 4, 8, 16, 32, 64, 128, 256, EHB, HAM, HAM8** or **CUST**.

The second form of the command allows the rendering type to be set to the supplied string which must match one of the above mentioned values.

If string was	RC	ADPRO_RESULT
accepted	0	the *previous* rendering type
not accepted	10	"Invalid Argument To Function"

For more information about rendering type, please see Section 3.5.5.

Execute

Syntax: **EXECUTE**

This command is the analog of the **Execute** button on the ADPro main screen. It causes rendered data to be created from raw data.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

For more information about the **Execute** button, please refer to Section 3.3.2.

Displaying a Rendered Image

Syntax: **ADPRO_DISPLAY**

This command is equivalent to the **ReDisplay** button on the ADPro main screen. Executing this command will cause any Amiga displayable rendered image to pop to front. Images rendered in the A-RES modes cannot be displayed using this command.

If	RC	ADPRO_RESULT
an image was displayed	0	undefined
only A-RES mode Amiga displayable data is available	10	"Error"

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Ending the Display of a Rendered Image

Syntax: **ADPRO_UNDISPLAY**

This command will move the screen containing any Amiga displayable image, which might be currently defined, to the backmost screen. The screen which will be revealed may not necessarily be the ADPro main screen. If this is required, you must issue an **ADPRO_TO_FRONT** as well.

Operation	RC	ADPRO_RESULT
always returns	0	undefined

ADPro Version

Syntax: **VERSION**

This command returns the version identification string of the currently executing ADPro.

Operation	RC	ADPRO_RESULT
always returns	0	version string

Serial Number

Syntax: **SERIAL_NUMBER**

This command returns the serial number of this copy of ADPro in **ADPRO_RESULT**. The serial number can be displayed by clicking on the **About** button on the main screen. This command will never return an error.

Operation	RC	ADPRO_RESULT
always returns	0	serial number

Determining Image Memory Size

Syntax: **CURRENT_MEM_SIZE**

This command will return the number of bytes of main memory in use by ADPro's primary image memory buffer. This value is returned in **ADPRO_RESULT**.

This value is reported from the user interface in the **About** panel.

Operation	RC	ADPRO_RESULT
always returns	0	image buffer size

Exiting ADPro

Syntax: **ADPRO_EXIT**

This command causes ADPro to terminate. Clearly, any ARexx command destined for ADPro executed after this command will not work, since ADPro will no longer be present to service it.

Operation	RC	ADPRO_RESULT
always returns	0	unchanged

Bringing the ADPro Screen to Front

Syntax: **ADPRO_TO_FRONT**

This command causes the ADPro main control screen to pop in front of all other screens.

Operation	RC	ADPRO_RESULT
always returns	0	undefined

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Pushing the ADPro Screen Behind

Syntax: `ADPRO_TO_BACK`

This command causes the ADPro main control screen to be pushed behind all other screens.

Operation	RC	ADPRO_RESULT
always returns	0	undefined

10.2.2 Balancing Controls

ARexx Interface

These commands interrogate or set values affecting color balancing.

Red Adjustment

Syntax: `RED` (1)
`RED` *value* (2)

The first form returns the current red adjustment setting.

The second format sets the red adjustment to the supplied value, which must be in the range of -50 to 50.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* red adjustment setting
out of range	10	"Invalid Argument To Function"

Note that if the value you wish to specify is a negative number, you must surround the negative number with quotation marks.

See Section 3.4.1 for more information about the red adjustment.

Green Adjustment

Syntax: `GREEN` (1)
`GREEN` *value* (2)

The first form returns the current green adjustment setting.

The second format sets the green adjustment to the supplied value, which must be in the range of -50 to 50.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* green adjustment setting
out of range	10	"Invalid Argument To Function"

Note that if the value you wish to specify is a negative number, you must surround the negative number with quotation marks.

See Section 3.4.1 for more information about the green adjustment.

Blue Adjustment

Syntax: **BLUE** (1)
BLUE *value* (2)

The first form returns the current blue adjustment setting.

The second format sets the blue adjustment to the supplied value, which must be in the range of -50 to 50.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* blue adjustment setting
out of range	10	"Invalid Argument To Function"

Note that if the value you wish to specify is a negative number, you must surround the negative number with quotation marks.

See Section 3.4.1 for more information about the blue adjustment.

Brightness Adjustment

Syntax: **BRIGHTNESS** (1)
BRIGHTNESS *value* (2)

The first form returns the current brightness adjustment setting.

The second format sets the brightness adjustment to the supplied value, which must be in the range of - 50 to 50.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* brightness adjustment setting
out of range	10	"Invalid Argument To Function"

Note that if the value you wish to specify is a negative number, you must surround the negative number with quotation marks.

See Section 3.4.1 for more information about the brightness adjustment.

Contrast Adjustment

Syntax: **CONTRAST** (1)
CONTRAST *value* (2)

The first form returns the current contrast adjustment setting.

The second format sets the contrast adjustment to the supplied value, which must be in the range of -50 to 50.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* contrast adjustment setting
out of range	10	"Invalid Argument To Function"

Note that if the value you wish to specify is a negative number, you must surround the negative number with quotation marks.

See Section 3.4.1 for more information about the contrast adjustment.

Gamma Adjustment

Syntax: **GAMMA** (1)
GAMMA *value* (2)

The first form returns the current gamma adjustment setting.

The second format sets the gamma adjustment to the supplied value, which must be in the range of 0 to 100.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* gamma adjustment setting
out of range	10	"Invalid Argument To Function"

NOTE Although you can set a gamma value in the range of -50 to 50 in the user interface (the **Gamma** slider and input field), the ARexx interface must be specified in the range of 0 to 100, where 50 means no gamma correction.

See Section 3.4.1 for more information about the gamma adjustment.

10.2.3 Palette Controls

ARexx Interface

The commands in this subsection refer to and manipulate the settings in the Palette control panel.

Palette Width

Syntax: `PWIDTH` (1)
`PWIDTH (NORMAL | ENHANCED)` (2)

The first form of the command returns the current width (accuracy) of the palette in the `ADPRO_RESULT` variable.

The second form of the command allows the ARexx programmer to set the palette width to the supplied value, which must be either `NORMAL` or `ENHANCED`.

If value was	RC	ADPRO_RESULT
either `NORMAL` or `ENHANCED`	0	the *previous* palette depth
neither `NORMAL` nor `ENHANCED`	10	"Invalid Argument To Function"

Note then when the palette is locked and you switch from `ENHANCED` to `NORMAL`, some resolution is lost. Switching back does not regain this lost resolution. When the palette is unlocked, palette recomputation will be done at the current palette width.

For more information concerning palette width, please refer to Section 3.4.2.

Palette Status

Syntax: `PSTATUS` (1)
`PSTATUS (LOCKED | UNLOCKED)` (2)

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The first form returns the string **LOCKED** or **UNLOCKED** in the **ADPRO_RESULT** variable, provided that results have been requested.

The second form sets the status of the palette to the specified value, which must be the string **LOCKED** or **UNLOCKED**.

If value was	RC	ADPRO_RESULT
either **LOCKED** or **UNLOCKED**	0	the *previous* state of the palette
neither **LOCKED** nor **UNLOCKED**	10	"Invalid Argument To Function"

For more information about palette status, refer to Section 3.4.2.

Total Number of Colors

Syntax: **PTOTAL** (1)
PTOTAL *value* (2)

The first form returns the currently defined total number of colors in **ADPRO_RESULT** provided that results have been requested.

The second form sets the total number of colors to the supplied value which must be one of the following: **2, 4, 8, 16, 32, 64, 128, 256, EHB, HAM,** or **HAM8.**

When specifying **HAM**, the actual total number of colors possible refers to the number of color registers available in the standard **HAM** mode (which is 16). Similarly, when referring to an **EHB** screen, the actual total number of colors available is 32.

If value was	RC	ADPRO_RESULT
accepted	0	total number of colors
out of range	10	"Invalid Argument To Function"

Note that the total number of colors, the offset of color zero, and the number of colors to be used must be consistent. For example, an error will be generated if you request a total number of colors which is smaller than the currently defined number of colors used.

For more information about the total number of colors, please see Section 3.4.2.

Number of Colors Used

Syntax: **PUSED** (1)
PUSED *value* (2)

The first form returns the current number of colors to be used in **ADPRO_RESULT** provided that results have been requested.

The second form sets the number of colors to be used to the supplied value which must range between 2 and the total number of colors.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* number of colors used
out of range	10	"Invalid Argument To Function"

Note that the total number of colors, the offset of color zero, and the number of colors to be used must be consistent. For example, an error will be generated if you request a number of colors to be used which is larger than the total number of colors.

For more information about the number of colors to be used, please see Section 3.4.2.

Offset Color Zero

Syntax: **POFFSET** (1)
POFFSET *value* (2)

The first form returns the currently defined offset of color zero in **ADPRO_RESULT** provided that results have been requested.

The second form sets the offset of color zero to the supplied value, which must range from 0 to the total number of colors minus one. When in **HAM** mode, the maximum value for the offset of color zero is 14. When in **HAM8** mode, the maximum value for the offset of color zero is 62. When in **EHB** mode, the maximum is 30.

If value was	RC	ADPRO_RESULT
accepted	0	offset value
out of range	10	"Invalid Argument To Function"

Note that the total number of colors, the offset of color zero, and the number of colors to be used must be consistent. For example, an error will be generated if you request an offset of color zero which would overflow the total number of colors presently defined when the number of colors to be used is factored in.

For more information about the offset of color zero, please refer to Section 3.4.2.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Palette Contrast

Syntax: **PCONTRAST** (1)
PCONTRAST (0 | 1) (2)

The first form of the command returns the present setting of the palette contrast button. If **ADPRO_RESULT** contains a 0, then the palette contrast button is set to **No Cont**. Otherwise, the palette contrast button is set to **Cont**.

The second form allows you to set the state of the palette contrast button. If you supply an argument of 0, then the palette contrast button is set to the off or **No Cont** setting. If you supply an argument of 1, then the palette contrast button is set to the on or **Cont** setting.

If value was	RC	ADPRO_RESULT
either 0 or 1	0	undefined
neither 0 nor 1	10	undefined

For more information about the palette button, please refer to Section 3.4.2.

Palette Sort Order

Syntax: **PSORT** (1)
PSORT (0 | 1) (2)

The first form of the command returns the present palette sort order. If **ADPRO_RESULT** contains a 0, then the sort order is ascending. Otherwise, the sort order is descending.

The second form sets the palette sort order to ascending or descending depending upon whether the supplied argument is a 0 or 1, respectively.

If value was	RC	ADPRO_RESULT
either 0 or 1	0	the *previous* sort direction
neither 0 nor 1	10	"Invalid Argument To Function"

For more information concerning palette sorting, please refer to Section 3.4.2.

Setting/Getting Palette Contents

Syntax: **PPOKE** *register* (1)
PPOKE *register red green blue* (2)

The first form of the command returns a string describing the contents of the specified color register. The string, returned in **ADPRO_RESULT**, is formatted as the red, green, and blue values of the color register separated by spaces and provided in decimal in the range of 0 to 255.

The second form of the command allows you to set the contents of the specified color register with the supplied values. Range checking will be performed on the supplied color values, which should range from 0 to 255.

For both forms, range checking will be performed on the supplied register number.

If operation was	RC	ADPRO_RESULT
successful	0	*"red green blue"*
not successful	non-zero	string indicating the nature of the error

While the allowable range of color values is from 0 to 255 (24 bit accuracy), AD-Pro's internal storage of the colors you define may vary according to whether you are in a normal or enhanced palette mode. As a consequence, it is normal to read a slightly different value than what might have been written immediately before.

You can convert 12 bit colors (the ones you would use with Deluxe Paint, for example) into 24 bit colors by multiplying each color by 17. So, for example, 15 becomes 255 while 1 would become 17.

Getting the Workbench Palette

Syntax: **PGETWB**

This command attempts to load the currently defined Workbench palette.

Note that this command results in the loading of up to four colors only. Should you require access to a deeper Workbench palette (possible under Workbench 2.0 and later), you can use the **PLOAD** command instead.

The Workbench palette is loaded starting with the color register specified by **POFFSET**.

Operation	RC	ADPRO_RESULT
always returns	0	unchanged

For more information about this command, please refer to Section 3.4.2.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Loading a Palette

Syntax: **PLOAD** (1)
PLOAD *filename* (2)

If no filename is supplied (as in the first form), then ADPro will bring up its file requester so that the user can manually enter the name of a palette file to be loaded. Note that ADPro will automatically pop its screen to front whenever it displays its file requester. Therefore, it is the ARexx programmer's responsibility to depth-arrange screens after calling this function with no supplied filename.

The second form of this command attempts to load a palette from the named file. The supplied file must be in **IFF** format but does not have to contain any image data.

If palette was	RC	ADPRO_RESULT
loaded	0	empty string
not loaded	10	empty string

Note that loading a palette is directly affected by the total number of colors, the number of colors used, and the offset of color zero.

For more information concerning the loading of a palette, please see Section 3.4.2.

Saving a Palette

Syntax: **PSAVE** (1)
PSAVE *filename* (2)

If a filename is not supplied (as in the first form), ADPro will display its file requester to allow the user to input a filename manually. Remember to handle the case of the user not making any file selection at all.

The second form of this command attempts to save the currently defined palette using the provided filename. The resulting file will be encoded in the **IFF** format.

If palette was	RC	ADPRO_RESULT
saved	0	empty string
not saved	10	empty string

It is the ARexx programmer's responsibility to prevent the unwanted overwriting of a pre-existing file.

For more information about saving a palette, please refer to Section 3.4.2.

The Wrong Way to Control the Palette

In each of the definitions for **PTOTAL**, **PUSED**, and **POFFSET**, a warning is given that the total number of colors, the number of colors to be used, and the offset of color zero must be consistent.

For example, the following is one of many wrong ways to set the total number of colors to 16 with 5 used and a color zero offset of 3:

```
POFFSET 3
PTOTAL 16
POFFSET 3
PUSED 5
```

The above example will fail if the number of colors to be used had an initial value above 12. This is because setting the offset of color zero to 3 would exceed the total number of colors if the number of colors to be used was 13 or higher.

Pathological cases abound, where the order in which these values are set will determine if an error occurs or not. Setting the offset of color zero to zero (then setting the other desired values) always increases your safety.

Make use of the query versions of the palette commands prior to setting palette values to avoid running afoul of ADPro's internal consistency check ($PUSED + POFFSET \leq PTOTAL$).

10.2.4 Screen Controls

ARexx Interface

Screen Type

Syntax: **SCREEN_TYPE** (1)
SCREEN_TYPE *mask* (2)

The first form of the command returns the current screen type in **ADPRO_RESULT**. The value returned is in the form of a bit-mask comprised of the values indicated in Table 10.1.

The second form of the command allows the screen type to be set to the supplied value. The supplied value must be within the range of 0 to 31.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Mode	Value
Hi Resolution On	1
Interlace On	2
PAL On	4
Horizontal Overscan On	8
Vertical Overscan On	16
VGA Mode On	32
Super Hi-Res Mode On	64
Super72 Mode On	128
Default Mode On	256

Table 10.1: Mask values for setting the screen type.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* screen type
out of range	10	"Invalid Argument To Function"

Your ARexx program might include the following:

```
HIRES = 1
INTERLACE = 2
PAL = 4
HOVERSCAN = 8
VOVERSCAN = 16
VGA = 32
SUPERHIRES = 64
SUPER72 = 128
DEFAULT = 256
```

Then, elsewhere in your ARexx program you can say things like:

```
SCREEN_TYPE HIRES + INTERLACE + HOVERSCAN + VOVERSCAN
```

This command would set the screen type to high resolution, NTSC, interlace, with both horizontal and vertical overscan.

Note that some modes cannot be used with some others. Specifically the flags for VGA, PAL, DEFAULT, and SUPER72 are mutually exclusive and the flags for Super Hi-Res and Hi-Res are mutually exclusive. If you specify these combinations, an error will result, which sets **ADPRO_RESULT** to **Invalid Argument To Function**.

ID	Method
0	None
1	Floyd-Steinberg
2	Burkes
3	Sierra
4	Jarvis
5	Stucki
6	Random
7	Large Ordered
8	Small Ordered

Table 10.2: Dither methods and their identifiers.

Another possible error condition is setting a legal combination of flags which simply aren't supported on the Amiga you are currently using. For example, asking for Super Hi-Res on a machine without Super Hi-Res capability will fail. The error returned in **ADPRO_RESULT**, which indicates that the specified flags were legal but that ADPro could not comply, is simply **ERROR**.

Dither Selection

Syntax: **DITHER** (1)
DITHER *method_number* (2)

The first form returns the identifier of the currently selected dither method. The dither methods currently defined are given in Table 10.2.

The second form sets the dither method to the method corresponding to the supplied value, which must be in the range of 0 to 8.

If value was	RC	ADPRO_RESULT
accepted	0	the *previous* dither method
out of range	10	"Invalid Argument To Function"

See Section 3.5.4 for more information about dithering.

Dither Amount

Syntax: **DITHER_AMOUNT** *amt*

This command lets you specify the amount of effect for the Random and Ordered dithers. The *amt* parameter must be between 1 and 256, with 1 having

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

the least effect and 256 having the most. If a dither other that Random or Ordered is specified, then this setting is effectively ignored.

10.2.5 Other ARexx Commands

This section describes commands which are available through ARexx which do not have direct equivalents in the ADPro user interface or don't fall into any of the previously described categories.

Last Saved Image

Syntax: `LAST_SAVED_IMAGE`

This command returns (in `ADPRO_RESULT`) the entire path name used during the last manually executed **Save** command. A manually executed **Save** is one *not* performed from ARexx.

If manually performed **Save** or **Load**	RC	ADPRO_RESULT
has been performed	0	path name
has not been performed	10	undefined

Note that if the user has not performed a **Save** by the time the first **Load** is executed, ADPro will copy the **Load** filename to the **Save** filename. This is the only circumstance in which `LAST_SAVED_IMAGE` will return a filename which wasn't actually saved.

Last Loaded Image

Syntax: `LAST_LOADED_IMAGE`

This command returns (in `ADPRO_RESULT`) the entire path name used during the last manually executed **Load** command. A manually executed **Load** is one *not* performed from ARexx.

If manually performed **Load** or **Save**	RC	ADPRO_RESULT
has been performed	0	path name
has not been performed	10	undefined

Note that if the user has not performed a **Load** by the time the first **Save** is executed, ADPro will copy the **Save** filename to the **Load** filename. This

is the only circumstance in which **LAST_LOADED_IMAGE** will return a filename which wasn't actually loaded.

Image Width

Syntax: **XSIZE**

This command returns the width of the currently loaded image in **ADPRO_RESULT**.

Operation	RC	ADPRO_RESULT
always returns	0	width of image

Image Height

Syntax: **YSIZE**

This command returns the height of the currently loaded image in **ADPRO_RESULT**.

Operation	RC	ADPRO_RESULT
always returns	0	height of image

Determining Currently Available Data

Syntax: **IMAGE_TYPE**

This command returns a string describing what kinds of data are available within ADPro at the present time. The string is returned in **ADPRO_RESULT** and can have the following forms:

NONE

This means that no image data (either raw or rendered) is currently available within ADPro.

COLOR

This means that raw color data is available but no rendered data is available at the current time.

GRAY

This means that raw grayscale data is available but no rendered data is available at the current time.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

BITPLANE

This means that rendered image data is available but that no raw color or grayscale data is presently available.

COLOR BITPLANE

This means that both raw color and rendered image data are currently available.

GRAY BITPLANE

This means that both raw grayscale and rendered image data are currently available within ADPro.

Operation	RC	ADPRO_RESULT
always returns	0	NONE, COLOR, GRAY, BITPLANE, COLOR BITPLANE, or GRAY BITPLANE

The ARexx programmer must not make any assumptions about the order in which the words in the result string will appear. Instead of direct string comparisons to determine the contents of the result string, use ARexx's substring functions.

Pausing a Specified Time

Syntax: **PAUSE** *ticks*

This command "kills time" for the specified number of ticks. A tick lasts for one-fiftieth of a second. The specified value must be in the range of 1 to 180000.

If value was	RC	ADPRO_RESULT
accepted	0	unchanged
out of range	10	"Invalid Argument To Function"

Getting a String From the User

Syntax: **GETSTRING** (1)
GETSTRING *title* (2)
GETSTRING *title default* (3)

This command can be used to query the user for a string to be returned into your ARexx program.

The first form of the command displays a string requester with a default title and no default text.

The second form uses the supplied string for the title bar of the string requester.

The third form uses the supplied strings for the title bar of the string requester and the default contents of the string requester, respectively.

If a string was	RC	ADPRO_RESULT
entered	0	the entered string
not entered	10	undefined

It is the ARexx programmer's responsibility to move ADPro to front (using **ADPRO_TO_FRONT**) before calling this function.

Note that strings with spaces must be quoted properly as in the following example:

```
GETSTRING 'Spaces' '"Require Double Quoting!"'
```

Getting a Number From the User

Syntax: **GETNUMBER** (1)
GETNUMBER *title* (2)
GETNUMBER *title default* (3)
GETNUMBER *title default min max* (4)

This command can be used to get a signed integer from the user.

The first form of the command displays an integer requester with a default title and no default value.

The second form uses the supplied title and no default value.

The third form uses the supplied title and the supplied default value.

The fourth form allows you to specify a minimum and maximum value between which the users input must fall.

If a number was	RC	ADPRO_RESULT
entered	0	the entered value
not entered	10	undefined

It is the ARexx programmer's responsibility to move ADPro to front (using **ADPRO_TO_FRONT**) before calling this function.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Note that strings with spaces must be quoted properly as in the following example:

```
GETNUMBER '"Spaces Require Double Quoting!"'
```

Getting a Filename From the User

Syntax:
`GETFILE` (1)
`GETFILE` *title* (2)
`GETFILE` *title default_dir* (3)
`GETFILE` *title default_dir default_file* (4)

This command can be used to get a filename from the user.

The first form of the command displays the file requester with a default title and no default filename.

The second form uses the supplied title and no default filename.

The third form uses the supplied title and the supplied default directory name. That is, the file requester will load the contents of the specified default directory.

The fourth form allows you to specify a title, a default directory, as well as a default filename.

If a file was	RC	ADPRO_RESULT
selected	0	the selected filename
not selected	10	undefined

It is the ARexx programmer's responsibility to move ADPro to front (using `ADPRO_TO_FRONT`) before calling this function.

Note that strings with spaces must be quoted properly as in the following example:

```
GETFILE '"Spaces Require Double Quoting!"'
```

Getting a List of Filenames From the User

Syntax:
`GETFILES` (1)
`GETFILES` *title* (2)

GETFILES *title default_dir* (3)
GETFILES *title default_dir default_file* (4)

This command can be used to get a list of filenames from the user.

The first form of the command displays the file requester with a default title and no default filename.

The second form uses the supplied title and no default filename.

The third form uses the supplied title and the supplied default directory name. That is, the file requester will load the contents of the specified default directory.

The fourth form allows you to specify a title, a default directory, as well as a default filename.

If	RC	ADPRO_RESULT
at least one file was selected	0	the selected filenames, separated by quotation marks
no file was selected	10	undefined

To select more than one file using the file requester, simply hold down either **Shift** key while clicking on filenames. All filenames selected must be from the same directory. This command makes it easier to write ARexx programs which will batch process multiple files, since it gives you a way of allowing the user to specify multiple files at one time.

It is the ARexx programmer's responsibility to move ADPro to front (using **ADPRO_TO_FRONT**) before calling this function.

Note that strings with spaces must be quoted properly as in the following example:

```
GETFILES '"Spaces Require Double Quoting!"'
```

Following, is an examples of using **GETFILES** and parsing its results.

```
/*
** An example of using GETFILES.
*/

ADDRESS "ADPro"
OPTIONS RESULTS

GETFILES
```

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

```
IF RC ~= 0 THEN DO
   /* No selection */
   EXIT
END

LIST = ADPRO_RESULT
COUNT = WORDS( ADPRO_RESULT )
I = 1

DO WHILE (I <= COUNT)
   OKAY1 WORD( LIST, I )
   I = I + 1
END
```

Getting a Directory Name From the User

Syntax: `GETDIR` (1)
`GETDIR` *title* (2)
`GETDIR` *title default* (3)

This command can be used to get a directory name or path from the user. This is especially handy when you wish to specify a directory in which your input (or output) frames may be found (or placed).

The first form of the command displays the file requester with a default title and no default filename.

The second form uses the supplied title and no default filename.

The third form uses the supplied title and the supplied default filename.

If a directory was	RC	ADPRO_RESULT
selected	0	the selected path
not selected	10	undefined

This command uses the same file requester as used elsewhere in this program. However, only directories will be shown to the user. Ordinary files will be ignored.

It is the ARexx programmer's responsibility to move ADPro to front (using `ADPRO_TO_FRONT`) before calling this function.

Note that strings with spaces must be quoted properly as in the following example:

```
GETDIR '"Spaces Require Double Quoting!"'
```

The OKAY1 Requester

Syntax: **OKAY1** "*string*"

The **OKAY1** command can be used to display an arbitrary string. The string will be rendered on the ADPro screen and execution will pause until the user selects the "resume" button in the **OKAY1** requester.

Operation	RC	ADPRO_RESULT
always returns	1	undefined

It is the ARexx programmer's responsibility to move ADPro to front (using **ADPRO_TO_FRONT**) before calling this function.

The OKAY2 Requester

Syntax: **OKAY2** "*string*"

The **OKAY2** command can be used to display an arbitrary string and solicit a two valued answer from the user. The string will be rendered on the ADPro screen and execution will pause until the user selects the "OK" or "Cancel" buttons in the **OKAY2** requester.

If the user selects	RC	ADPRO_RESULT
"Cancel"	0	undefined
"OK"	non-zero	undefined

It is the ARexx programmer's responsibility to move ADPro to front (using **ADPRO_TO_FRONT**) before calling this function.

Pixel Aspect

Syntax: **PIXEL_ASPECT**

This command returns the current pixel aspect ratio in **ADPRO_RESULT**.

If an image was	RC	ADPRO_RESULT
available	0	pixel aspect
not available	10	unchanged

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Pixel Resolution

Syntax: `PIXEL_RESOLUTION`

This command returns the current pixel resolution (horizontal and vertical DPI) in `ADPRO_RESULT`.

If an image was	RC	ADPRO_RESULT
available	0	pixel resolution
not available	10	unchanged

Obtaining Exclusive Use of ADPro

Syntax: `OBTAIN_ADPRO`

This command tries to place an exclusive "lock" on ADPro, so that your script will have sole access of the program. This is very important when two or more scripts are likely to call ADPro at the same time.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

Releasing Exclusive Use of ADPro

Syntax: `RELEASE_ADPRO`

This command will release the exclusive "lock" on ADPro, so that other scripts may be able to lock the program.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

Starting ADPro Via ARexx

The following ARexx fragment can be adapted for use in your own ARexx programs. This code attempts to locate a currently running ADPro. Failing that, it will attempt to start up ADPro. In order to do this, `ADPRO:` must be defined.

```
** $VER: Locate_ADPro.rexx 1.00 (1.9.92)
**
** This ARexx program will attempt to find
** a currently running ADPro. If one is
** not found, then it will attempt to start
** up ADPro.
**
** The main guts of this program are
** imbedded in a sub-routine to make it
** easier to glue into your own code.
**
** Example ARexx program for controlling
** ADPro by ASDG, Incorporated.
** Copyright 1992 ASDG, Incorporated.
** All Rights Reserved Worldwide
*/

OPTIONS RESULTS

CALL Locate_ADPro

IF RESULT = 1 THEN
   SAY "ADPro has been found"
ELSE
   SAY "Could not locate or start ADPro"
EXIT


Locate_ADPro:

Max_Seconds_To_Load = 60
Flag = 0
LibName = 'rexxsupport.library'

IF POS(LibName, SHOW('Libraries')) = 0 THEN
   ADDLIB(LibName, 0, -30, 0)
IF POS(LibName, SHOW('Libraries')) = 0 THEN
   RETURN 0

IF STATEF('ADPRO:') = "" THEN
   RETURN 0

TIME('R')
DO WHILE ((TIME('E') < Max_Seconds_To_Load) &
          (POS('ADPro', SHOW('Ports')) = 0))
   IF Flag = 0 THEN DO
```

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

```
      ADDRESS COMMAND 'run < nil: > nil: ADPRO:ADPro'
      Flag = 1
   END
   ADDRESS COMMAND 'WAIT 1'
END
IF POS('ADPro', SHOW('Ports')) = 0 THEN
   RETURN 0
ELSE
   RETURN 1
```

10.3 Loaders

Loaders can be selected from the main screen or through their ARexx interfaces. For specific information, read each loader's control panel and ARexx interface sections.

10.3.1 General Information

ARexx Interface

Load Format

Syntax: `LFORMAT` (1)
`LFORMAT` *format* (2)

The first form of the command will fill the `ADPRO_RESULT` field with the name of the currently selected Load Format, if results are requested.

The second form of the command sets the Load Format to the specified string.

If module	RC	ADPRO_RESULT
exists	0	the *previously* selected Load Format
does not exist	10	"Unknown Module"

Refer to Section 2.3.1 and Chapter 4 for more information concerning Load Formats.

Calling a Loader

Syntax: **LOAD** *filename* [*loader_opts*] [*comp_opts*] [**FORCE_PALETTE**]

This command causes the specified file to be loaded. The only required argument is the filename to be loaded. If you wish to allow the user to manually enter a filename to load, then you must supply a NULL *filename* (in ARexx, this would be: '"' || '"').

The loader which will be invoked is specified with the **LFORMAT** command.

The *loader_opts* arguments are loader dependent. The currently defined loader-specific arguments are defined in each loader's ARexx Interface reference section.

comp_opts define the ARexx interface to the image compositing capability of ADPro. *comp_opts* may be specified in one of two formats (old and new).

Old Style

left_edge top_edge mix_level (1)
left_edge top_edge mix_level rt gt bt (2)

In both forms, you must specify where the image to be loaded will be placed relative to the upper left-hand corner of the image currently in memory. This is specified by *left_edge* and *top_edge*. Also, both forms require that you specify the mix level to be used (which may range in value from 1 to 100).

The second allows you to specify which 24 bit-plane color value will be considered transparent. If you are merging a grayscale image, the gray level to be considered transparent is taken to be *rt*. In the first form, transparency is disabled.

New Style

The new style compositing arguments can be placed anywhere on the argument line after the *filename* argument. Zero or more of the following can be defined:

COMPOFFSET *left top*
Specify the image offset of the foreground image relative to the background image.

GUI Equivalent: **Off X** and **Off Y** values.
Constraints: None
Required: No

COMPMIX *mix*
Specify the mix percentage.

GUI Equivalent: **Mix** value.
Constraints: $0 \leq mix \leq 100$
Required: No

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

COMPTRANS *r g b*

Specify the transparent color (in the foreground image).

GUI Equivalent:	**R**, **G**, and **B** values.
Constraints:	$-1 \leq r \leq 255$; $-1 \leq g \leq 255$; $-1 \leq b \leq 255$
Required:	No

NOPAD

Specify that raw data should not be created.

GUI Equivalent:	None
Constraints:	None
Required:	No

For more information about image compositing, please refer to Section 2.3.4.

The **FORCE_PALETTE** argument allows the loaded image's palette (if it has one) to replace the current palette even if the palette is currently locked. The **FORCE_PALETTE** option, if desired, must be the last option specified.

If the load operation is successful, **RC** = 0.

Some examples follow:

```
LOAD Filename
```

This simply loads the file specified by the variable **Filename**. This image completely replaces any previous image in memory.

```
LFORMAT "IFF"
LOAD Filename 0 0 100 255 255 255
```

This shows an example of loading an IFF-ILBM file with the image compositing options present. This will load the image at an offset of 0,0. The mix level is 100 percent and pure white will be considered to be transparent.

Load Orientation

Syntax: **ORIENTATION** (1)
ORIENTATION (PORTRAIT | LANDSCAPE) (2)

The first form of the command returns the current load orientation. The **ADPRO_RESULT** variable will be set to either **PORTRAIT** or **LANDSCAPE**.

The second and third forms of the command set the load orientation to the specified value.

For more information about the uses of portrait and landscape orientations, please refer to Section 2.3.2.

Syntax: **LOADER** *loader_module filename loader_args*

An alternative to the **LFORMAT** ... **LOAD** sequence is the **LOADER** command. It takes both the name of the loader and its arguments on one line. This is a very useful time-saver in that you can set and call a loader using one command (**LOADER**). Note that you must specify a *filename* as well as a module name (*loader_module*).

10.3.2 ALPHA

The ALPHA loader lets you use a grayscale IFF-ILBM image as the alpha channel when compositing an image over the current image in ADPro. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "ALPHA"**
LOAD *image_to_load loader_opts*

The only *loader_opts* for the ALPHA loader is:

alpha_channel_file
: The name of the grayscale IFF-ILBM file to use as the alpha channel.

The *comp_opts* arguments (old format) must follow the *loader_opts* above.

Be sure you do not delete the **.ALPHA** and **ALPHA** files from the **Loaders2** directory. The ALPHA loader needs both of these files to operate. These files are placed in this directory by the installation program.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

10.3.3 ANIM

The ANIM loader lets you load a single frame from IFF-ANIM animation files. It can be accessed through the main control screen and through ARexx.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

ARexx Interface

Syntax: **LFORMAT "ANIM"**
LOAD *anim_filename loader_opts*

The *loader_opts* for the ANIM loader are:

(**FRAME** *n* | **COUNT** | **QUIT**)
Specify whether to load a specific frame (**FRAME**), count how many frames are available, or close the ANIM file so that other programs may use it.

GUI Equivalent: **Frame Number:** for **FRAME** *n*;
Count Frames for **COUNT**; **Quit** button.
Constraints: $0 \leq n$
Required: Yes

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	Error message

10.3.4 BACKDROP

The BACKDROP loader lets you create filled or gradated color or grayscale backgrounds. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "BACKDROP"**
LOAD "XXX" *loader_opts*

The BACKDROP loader's *loader_opts* must conform to one of the following:

width height type (1)
width height type red green blue (2)
width height type ulr ulg ulb urr urg urb llr llg llb lrr lrg lrb (3)

In each form, width and height specify the width and height of the bit-map to be created. A type of **"COLOR"** instructs ADPro to create a 24 bit-plane raw

color bit-map. A type of "GRAY" causes ADPro to create an 8 bit-plane raw gray scale bit-map of the specified size.

In the first form, the bit-map will be filled with black.

In the second form, the bit-map will be filled with the specified color.

In the third form, the bit-map will be filled with a gradated pattern formed from a linear interpolation between the colors specified as the four corners.

For grayscale bit-maps, colors are always specified as RGB triples in the range of 0 to 255. However, only the second (the green component) is used.

If backdrop was	RC	ADPRO_RESULT
created	0	undefined
not created	non-zero value	undefined

10.3.5 BACKLINE

The BACKLINE loader lets you create filled or gradated color or grayscale backgrounds. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: `LFORMAT "BACKLINE"`
`LOAD "XXX"` *loader_opts*

The BACKLINE loader's *loader_opts* must conform to the following:

type width height pos Start Middle End direction

where:

type
Can be either "COLOR" or "GRAY".

width
Is the width of the area to be created.

height
Is the height of the area to be created.

pos
Is the position of the middle color, relative to the start and end colors. This value may range from 0 to 100.

Start

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If *type* is set to "COLOR", then this is the starting color expressed as a red, green, and blue value (i.e., three integers between 0 and 255). If *type* is set to "GRAY", then this is the starting color expressed as a single gray level intensity (ranging from 0 to 255).

Middle

If *type* is set to "COLOR", then this is the middle color expressed as a red, green, and blue (i.e., three integers between 0 and 255). If *type* is set to "GRAY", then this is the middle color expressed as a single gray level intensity (ranging from 0 to 255).

End

If *type* is set to "COLOR", then this is the ending color expressed as a red, green, and blue value (i.e., three integers between 0 and 255). If *type* is set to "GRAY", then this is the ending color expressed as a single gray level intensity (ranging from 0 to 255).

direction

Is the direction of the ramp. This can be one of the following: "NW", "N", "NE", "E", "SE", "S", "SW", or "W".

These arguments must all be present. Composition options may follow these arguments.

Here's an example of how you might use the BACKLINE loader from ARexx:

```
/* Setting  up these variables makes the actual call
** to the BACKLINE loader much more readable.
*/

S = "200 100 50"
M = "50 200 100"
E = "100 50 200"
GRAY = "128 255 128"

LFORMAT "BACKLINE"

/* Here's a color backline call */

LOAD "XXX" "COLOR" 320 200 50 S M E "S"
IF RC ~= 0 THEN DO
     OKAY1 "BackLine Failed"
     EXIT
END

/* Here's a gray backline call.  Note that GRAY is a variable
** with three numbers in it.  But "GRAY" is the string ''GRAY''
** (i.e., the double quotes around "GRAY" mean--this is a string
** not a variable.)
```

```
*/

LOAD "XXX" "GRAY" 320 200 50 GRAY "W"
IF RC ~= 0 THEN DO
     OKAY1 "BackLine Failed"
     EXIT
END
```

If backline was	RC	ADPRO_RESULT
created	0	undefined
not created	non-zero value	undefined

10.3.6 BMP

The BMP loader lets you import images in the BMP format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "BMP"**
LOAD *filename loader_opts*

The only BMP loader *loader_opts* is **NOPAD**, which disables the conversion of rendered image data back into 24 or 8 bit-plane raw data during loading.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

10.3.7 CLIPBOARD

The CLIPBOARD loader lets you import image data from the Amiga's Clipboard. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "CLIPBOARD"**
LOAD "XXX" *loader_opts*

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The CLIPBOARD loader supports the **NOPAD** keyword for *loader_opts*. If the image in the Clipboard contains rendered image data (i.e., neither 24 bit-plane color nor 8 bit-plane gray), the **NOPAD** argument tells ADPro *not* to convert the rendered image data back into raw image data. Because this option disables the conversion on the newly loaded image into 24 bit-plane color or 8 bit-plane grayscale data, the image is loaded more quickly.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

10.3.8 DPIIE

The DPIIE loader lets you import images in the DPIIE format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "DPIIE"**
LOAD *filename*

No loader options are used by the DPIIE loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.9 DV21

The DV21 loader lets you import 21 bit-plane images saved by NewTek's Digi-View 3.0 *only*. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "DV21"**
LOAD *filename*

No loader options are used by the DV21 loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.10 FRAMEGRABBER

The FRAMEGRABBER loader lets you digitize images through the Progressive Peripherals, Inc. FrameGrabber video digitizer. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: `LFORMAT "FRAMEGRABBER"`
`LOAD "XXX"` *type*

where *type* can be one of:

`FIELD1`
: This causes the loader to download only the first field grabbed. This results in an image which measures 320 by 200 pixels in size.

`FIELD2`
: This causes the loader to download only the second field grabbed. This results in an image which measures 320 by 200 pixels in size.

`STACKED`
: This causes the loader to download both of the grabbed fields. Field 1 will be stacked above field 2. This option results in an image which measures 320 by 400 pixels in size.

`INTERLACED`
: This causes the loader to download both of the grabbed fields which are interlaced together to form one image. This option results in an image which measures 320 by 400 pixels in size.

If no grab type is specified (allowable only if you are not specifying any compositing options, then `INTERLACED` is assumed.

NOTE If you intend to specify compositing options, then you **must** specify a grab type!

It is not reasonable to expect to use the FRAMEGRABBER loader from ARexx to grab frames in a timing dependent fashion because there is no precise way of predicting exactly when the loader will actually grab the frame. The intended method for using this loader from ARexx is to set up the image to be grabbed (perhaps by issuing commands to a VCR or VideoDisc player), allow time for the image to become stable, and then execute the FRAMEGRABBER loader.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

NOTE The FRAMEGRABBER loader does not operate correctly on PAL machines.

If load was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.11 GIF

The GIF loader lets you import images in the GIF format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: `LFORMAT "GIF"`
`LOAD` *filename*

No loader options are used by the GIF loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.12 HAME

HAM-E

The ~~GIF~~ loader lets you import images in the HAM-E format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: `LFORMAT "HAME"`
`LOAD` *filename*

No loader options are used by the HAME loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.13 IFF

The IFF loader lets you load IFF-ILBM images. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "IFF"**
LOAD *image_to_load loader_opts*

The IFF loader supports the **NOPAD** keyword for *loader_opts*. If the IFF-ILBM file being read contains rendered image data (i.e., neither 24 bit-plane color nor 8 bit-plane gray), the **NOPAD** argument tells ADPro *not* to convert the rendered image data back into raw image data. Because this option disables the conversion on the newly loaded image into 24 bit-plane color or 8 bit-plane grayscale data, the image is loaded more quickly. This might be used from ARexx to implement a simple "slide-show" type program. This feature also makes conversions between same depth file formats much faster.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

Note that when **ADPRO_RESULT** = 5, it means that something was loaded, but not converted into raw image data. Therefore you shouldn't check **RC** this way:

```
/* WRONG WAY */
LFORMAT "IFF"
LOAD "SOMETHING" NOPAD
IF RC ~= 0 THEN DO
   OKAY1 "FAILED!"
   EXIT
END
```

but rather:

```
/* RIGHT WAY */
LFORMAT "IFF"
LOAD "SOMETHING" NOPAD
```

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

```
IF RC = 10 THEN DO
   OKAY1 "FAILED!"
   EXIT
END
```

10.3.14 IMPULSE

The IMPULSE loader lets you import images in RGBN or RGB8 format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "IMPULSE"**
LOAD *filename*

No loader options are used by the IMPULSE loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.15 IV24

The IV24 loader lets you grab images with Great Valley Product's IV24 framegrabber capabilities. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "IV24"**
LOAD "XXX" *loader_opts*

The IV24 loader's *loader_opts* must conform to one the following:

no options specified (1)
width height (2)
width height xoff yoff (3)

In the first form, the width, height, and offsets are taken from default values.

In the second form, the width and height are specified, while the offsets are taken from defaults.

In the third form, the width, height, and offsets are all specified.

An error is returned if the *width* plus the x offset (*xoff*) is larger than 768. An error is also returned if the *height* plus the y offset (*yoff*) is larger than 566. In both cases, the error is specified by an **RC** value of 10. **ADPRO_RESULT** is undefined after an error.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.16 JPEG

The JPEG loader lets you decompress images previously compressed in JPEG (JFIF) format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "JPEG"**
LOAD *filename loader_opts*

The only JPEG loader *loader_opts* option is:

SMOOTHING
Cause a smoothing operation to be performed while the image is being decompressed.

GUI Equivalent: **Smoothing:** cycle gadget set to **ON**.
Constraints: None
Required: No

If load was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.17 MACPAINT

The MACPAINT loader lets you import images in the MacPaint format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "MACPAINT"**
LOAD *filename*

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

No loader options are used by the MACPAINT loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.18 PCX

The PCX loader lets you import images in the PCX format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "PCX"**
LOAD *filename loader_opts*

The PCX loader supports the **NOPAD** keyword for *loader_opts*. If the PCX file being read contains rendered image data (i.e., neither 24 bit-plane color nor 8 bit-plane gray), the **NOPAD** argument tells ADPro *not* to convert the rendered image data back into raw image data. Because this option disables the conversion on the newly loaded image into 24 bit-plane color or 8 bit-plane grayscale data, the image is loaded more quickly. This might be used from ARexx to implement a simple "slide-show" type program. This feature also makes conversions between same depth file formats much faster.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

10.3.19 POINTER

The POINTER loader lets you grab the currently active pointer imagery. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "POINTER"**
LOAD "XXX"

The POINTER loader can be called from ARexx and will *pause the ARexx program until the correct hot-key sequence is depressed on the keyboard.* The correct sequence is **Right Alt + Right Shift + Backspace.**

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.20 QRT

The QRT loader lets you import images in the QRT format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "QRT"**
LOAD *filename*

No loader options are used by the QRT loader.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.21 SCREEN

The SCREEN loader lets you grab almost any current Amiga screen. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "SCREEN"**
LOAD *screen_name*

The *screen_name* argument specifies which screen is to be grabbed. The specified string is used as a prefix when matching against the Intuition names of each defined screen. By allowing the name you supply to be used as a prefix, the SCREEN loader can easily grab screens with very long screen names or which have screen names which change over time.

For example, ASDG's high performance editor, CygnusEd Professional (CEDPro), has a very long Intuition screen name which includes status information that changes over time. CEDPro screens always start with the characters,

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

CED. Therefore, specifying only CED is enough to cause a CEDPro screen to be grabbed.

Specifying a NULL ("") file name causes the frontmost screen to be grabbed. In this way, screens with no Intuition screen name can be grabbed. The ARexx program would first have to arrange for the nameless screen to be popped-to-front.

Specifying a *screen_name* of "?" does not cause a screen to be grabbed. Rather, it returns a list of the screens (having names) which are currently defined. This list will be returned in the ADPRO_RESULT variable. Each screen in the list will be surrounded by a double quote.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.22 SCULPT

The SCULPT loader lets you import images in the SCULPT format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: `LFORMAT "SCULPT"`
`LOAD` *filename loader_opts*

The SCULPT loader's *loader_opts* must conform to the following:

width height type

where:

width
Is the width of the image to be loaded.

height
Is the height of the image to be loaded.

type
Can be either "COLOR" or "GRAY".

These arguments must all be present. Composition options may follow these arguments.

NOTE From ARexx, you may specify only one file on the LOAD statement. If loading a color image, this file name must correspond to the

red component, which must end in the three letters, "**red**". In this case, the files to be loaded must end in the conventional extensions. When loading a grayscale image, you must completely specify the name of the single file to be loaded.

If load was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.3.23 TEMP

The TEMP loader lets you replace the raw image data in the primary image buffer with the raw image data currently in the temporary buffer. This loader can be executed from the main cotrol screen or through its ARexx interface.

ARexx Interface

Syntax: **LFORMAT "TEMP"**
LOAD "XXX"

There are no *loader_opts* for the TEMP loader.

The *comp_opts* arguments (new format) must follow the **"XXX"** parameter.

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	undefined

10.3.24 UNIVERSAL

The UNIVERSAL loader will try to load in a particular image file, using its filename and embedded information to detect and invoke the correct loader. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **LFORMAT "UNIVERSAL"**
LOAD *image_to_load*

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The UNIVERSAL loader requires some special discussion. You can use the UNIVERSAL loader from ARexx. However, since you don't know which loader will ultimately be used to load the image, you cannot make use of any loader options.

If load was	RC	ADPRO_RESULT
successful (not using NOPAD)	0	undefined
successful (using NOPAD)	5	undefined
unsuccessful (either case)	10	undefined

10.4 Savers

Savers can be selected from the main screen or through their ARexx interfaces. For specific information, read each saver's control panel and ARexx interface sections.

10.4.1 General Information

ARexx Interface

Save Format

Syntax: `SFORMAT` (1)
`SFORMAT` *format* (2)

The first form of the command fills the `ADPRO_RESULT` field with the name of the currently selected Save Format, if results are requested.

The second form of the command sets the Save Format to the specified string.

If module	RC	ADPRO_RESULT
exists	0	the *previously* selected Save Format
does not exist	10	"Unknown Module"

Refer to Section 2.5.1 and Chapter 5 for more information concerning Save Formats.

Calling a Saver

Syntax: **SAVE** *filename type [saver_opts]*

This command invokes a Saver with the specified arguments.

The Saver which will be invoked is specified with the **SFORMAT** command.

A filename and save type are required arguments. The type must be one of the following strings: **RAW**, **IMAGE**, or **SCREEN**. Specifying **RAW** will attempt to save the raw image data, if available. Specifying **IMAGE** will attempt to save the rendered image data, if available.

Specifying **SCREEN** will attempt to save only as much rendered image data as will fit in the currently defined screen type (and, will save only a screen sized portion of the image if it is larger than the current screen type). Screen size is defined by the currently set screen width and height. Also, the upper left hand corner of the portion of the image to be saved is determined by scrolling the image by hand. From ARexx, you cannot automatically offset the portion of the image to be saved.

Each particular saver may have its own restrictions as to which type of data may be saved. These restrictions are defined in each saver's ARexx Interface reference section.

Remember that if you only have rendered image data, you can turn it back into raw image data by saving the rendered image data to disk and then reloading it.

The *saver_opts* are saver specific options. The currently defined saver-specific arguments are defined in each saver's ARexx Interface reference section.

Execute a Saver Module

Syntax: **SAVER** *saver_module filename saver_args*

An alternative to the **SFORMAT** ... **SAVE** sequence is the **SAVER** command. It takes both the name of the saver and its arguments on one line. This is a very useful time-saver in that you can set and call a saver using one command (**SAVER**). Note that you must specify a *filename* as well as a module name (*saver_module*).

10.4.2 A2410

The A2410 saver lets you send image data to the A2410 display board. It can be accessed through the main control screen and through ARexx.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Resolution	Interlaced	ID
1024 x 768	No	0
800 x 600	No	1
640 x 400	Yes	2
1024 x 768	Yes	3
1024 x 1024	No	4
736 x 480	No	5
1024 x 1024	No	6

Table 10.3: TIGA screen IDs which may be used as arguments to the B_SIZE parameter.

ARexx Interface

Syntax: **SFORMAT "A2410"**
SAVE *filename type saver_opts*

The *type* argument must be **"IMAGE"** only, or **"RAW"** for 8-bit grayscale images.

The ARexx support for Commodore's A2410 display board constitutes a language in-and-of-itself. Each call to the A2410 saver from ARexx defines a stream of commands (defined as *saver_opts*) which are parsed and executed from left to right. These commands are:

BOARD_NO *num*
: If you have more than one A2410 installed, you can use this command to select which board subsequent commands will be sent to. All commands described here operate on the currently selected board.

B_SIZE *id*
: This command sets display resolution to match the TIGA screen *id* as given in Table 10.3. If the crystal matching the specified *id* is unavailable (i.e., not installed on the board), then the command will fail.

CENTER
: This command sets the display offsets to center the image currently in Amiga memory on the currently selected A2410 display.

CLEAR
: This command clears the currently selected board.

IMAGE
: This command causes the image currently in Amiga memory to be downloaded to the currently selected board at the current offsets.

SET_DOX *left_edge*

This command specifies which column (**DOX** means "Destination Offset X") the image will be written to.

SET_DOY *top_edge*
This command specifies which row (**DOY** means "Destination Offset Y") the image will be written to.

SET_DWX *width*
This command sets the width of the image area to be written to the specified value. The required argument *width* is in A2410 pixels.

SET_DWY *height*
This command sets the height of the image area to be written to the specified value. The required argument *height* is in A2410 pixels.

SET_SOX *left_edge*
This command specifies the left edge of the image area *within the image* in Amiga memory which will be written to the board (**SOX** means "Source Offset X").

SET_SOY *top_edge*
This command specifies the top edge of the image area *within the image* in Amiga memory which will be written to the board (**SOY** means "Source Offset Y").

Since ARexx command lines invoking the A2410 saver can be complex, finding an error on the command line can become difficult. The saver, therefore, returns in **ADPRO_RESULT** the number of the argument in which an error was found. The argument numbering begins with 1, which corresponds to the *type* parameter of the ARexx **SAVE** command. If an error is encountered, **RC** will return non-zero.

For instance, the following returns a 1 in **ADPRO_RESULT**:

```
SAVE "XXX" "NEITHER_IMAGE_NOR_SCREEN" "IMAGE"
```

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	parameter error number	undefined

10.4.3 ANIM

The ANIM saver can be controlled through a control panel or ARexx.

ARexx Interface

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Syntax: **SFORMAT "ANIM"**
SAVE *anim_filename saver_opts*

The ANIM saver may be instructed to save either **IMAGE** or **SCREEN** data (specified as one of the *saver_opts*).

The ANIM saver's *saver_opts* include:

(IMAGE | SCREEN)
Set the type of rendered data to save.

GUI Equivalent: **Save Type** cycle gadget value.
Constraints: None
Required: Yes

APPEND | WRAPUP | QUIT
Describe the operation to perform. **APPEND** adds the current rendered data to the end of the ANIM. **WRAPUP** appends the first two frames to the end of the ANIM *and* closes the file. **QUIT** closes the file so that other programs can access it.

GUI Equivalent: **Append**, **Wrap Up**, and **Quit** buttons on the control panel.
Constraints: None
Required: Yes

TRUNCATE *n*
TRUNCATE (takes an integer argument) either reduces the file to n frames (for positive n) or chops $-n$ frames (for negative n).

GUI Equivalent: **Truncate** button on the control panel. The n value is equivalent to the value specified in the Truncate control panel's input field.
Constraints: $0 \leq n$
Required: No

If load was	RC	ADPRO_RESULT
successful (not using **NOPAD**)	0	undefined
successful (using **NOPAD**)	5	undefined
unsuccessful (either case)	10	Error message

10.4.4 BMP

The BMP saver lets you export images in BMP format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "BMP"**
SAVE *filename type*

The *type* argument must be either **"RAW"** or **"IMAGE"**.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.5 CLIPBOARD

The CLIPBOARD saver lets you post image data to the Amiga's Clipboard. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "CLIPBOARD"**
SAVE *filename type*

The *type* argument must be **"RAW"**, **"IMAGE"**, or **"SCREEN"**.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.6 DPIIE

The DPIIE saver lets you export images in DPIIE format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "DPIIE"**
SAVE *filename type*

The *type* argument must be **"IMAGE"** only.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.4.7 FC24

The FC24 saver lets you export images to the FireCracker 24 display board. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "FC24"**
SAVE *filename type saver_opts*

The *type* argument must be either **"RAW"** or **"IMAGE"**.

The ARexx support for Impulse's FireCracker 24 display board constitutes a language in-and-of-itself. Each call to the FC24 saver from ARexx defines a stream of commands (defined as *saver_opts*) which are parsed and executed from left to right. These commands are:

AMIGA (ON | OFF)
This command turns the Amiga display on or off as specified. This is useful when the Amiga and the FireCracker 24 are sharing the same monitor.

BOARD (ON | OFF)
This command turns the currently selected FireCracker 24 display on or off as specified. This is useful when the Amiga and the FireCracker 24 are sharing the same monitor.

BOARD_NO *num*
If you have more than one FireCracker 24 installed, you can use this command to select which board subsequent commands will be sent to. All commands described here operate on the currently selected board.

B_WIDTH (384 | 512 | 768 | 1024)
This command sets display width to the specified value. Note two things: First, that changing the board width while the board is "on" necessarily causes that board's display to change. Second, the FireCracker 24's double buffering features can be used only when the board width is set to 384 or 512.

CENTER
This command sets the display offsets to center the image currently in Amiga memory.

CLEAR
This command clears the currently selected board or, if the display width is 512 or less, clears the currently selected buffer (as defined by **IMAGE_TO**).

DISPLAY_FROM (A | B)

If the currently selected board's width is 512 or less, then you can take advantage of the FireCracker 24's double buffering capability. This command specifies which of the two buffers will be displayed from and causes the switch to take place immediately.

IMAGE
This command causes the image currently in Amiga memory to be downloaded to the currently selected board and buffer at the current offsets.

IMAGE_TO (A | B)
When double buffering is possible, this command selects which buffer will be written to by a subsequent **IMAGE** command.

SET_DOX *left_edge*
This command specifies which column (**DOX** means "Destination Offset X") the image will be written to.

SET_DOY *top_edge*
This command specifies which row (**DOY** means "Destination Offset Y") the image will be written to.

SET_DWX *width*
This command sets the width of the image area to be written to the specified value. The required argument *width* is in FireCracker 24 pixels.

SET_DWY *height*
This command sets the height of the image area to be written to the specified value. The required argument *height* is in FireCracker 24 pixels.

SET_SOX *left_edge*
This command specifies the left edge of the image area *within the image* in Amiga memory which will be written to the board (**SOX** means "Source Offset X").

SET_SOY *top_edge*
This command specifies the top edge of the image area *within the image* in Amiga memory which will be written to the board (**SOY** means "Source Offset Y").

Since ARexx command lines invoking the FireCracker 24 saver can be complex, finding an error on the command line can become difficult. The saver, therefore, returns in **ADPRO_RESULT** the number of the argument in which an error was found. The argument numbering begins with 1, which corresponds to the *type* argument of the ARexx **SAVE** command. If an error is encountered, **RC** will return non-zero.

For instance, the following returns a 1 in **ADPRO_RESULT**:

```
SAVE "XXX" "NEITHER_RAW_NOR_IMAGE" "IMAGE"
```

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If you request a data type (either **RAW** or **IMAGE**) which is not available, **RC** will be set to 10, and **ADPRO_RESULT** will be set to 1.

For an example of how the FireCracker 24 saver is used from ARexx, please see **Example FG and FC24 Use** in Appendix B.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	1

10.4.8 FRAMEBUFFER

The FRAMEBUFFER saver lets you send images to the Mimetics FrameBuffer. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "FRAMEBUFFER"**
SAVE *filename type*

The *type* argument must be **"RAW"** only.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.9 GIF

The GIF saver lets you export images in GIF format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "GIF"**
SAVE *filename type*

The *type* argument must be **"IMAGE"** only.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.10 HAME

The HAME saver can be controlled through a control panel or ARexx.

ARexx Interface

Syntax:
```
SFORMAT "HAME"
SAVE filename "IMAGE" "TODISK"                          (1)
SAVE filename "IMAGE" "TODISK" screen_type              (2)
SAVE "XXX" "IMAGE" "DISPLAY"                            (3)
SAVE "XXX" "IMAGE" "DISPLAY" ticks                      (4)
SAVE "XXX" "IMAGE" "DISPLAY" ticks screen_type          (5)
```

The HAME saver can only save **IMAGE** data.

The first form saves the currently rendered HAME data to the file specified by *filename*. If *filename* is NULL, then the file requester will pop up to allow the user to manually select the name of the file in which to save the data. The screen settings used will be what ever had been last used.

The second form saves the currently rendered HAME data to the file specified by *filename*. If *filename* is NULL, then the file requester will pop up to allow the user to manually select the name of the file in which to save the data. The number specified by *screen_type* is used to determine what type of screen to construct. This number can be assembled using the values contained in Table 10.1. Note that the VGA and Super Hi-Res modes are not allowed.

The third form allows you to display the image data, rather than save it to disk. If the **"DISPLAY"** keyword is present, then the supplied filename is ignored. The image is displayed until the user clicks the left mouse button. The screen settings used to construct the display will be the same ones last used.

The fourth form will display the image data, but for a specific amount of time. In this form, you can specify the number of *ticks* (fiftieths of a second) to hold the image on-screen. The value specified by *ticks* may be either 0 or between 1 and 65535. If the value is 0, then the displayed image will remain on-screen until the left mouse button is clicked.

The fifth form will display the image data on a specific type of screen for a specific amount of time. The *ticks* argument is the same value as described in the fourth form. The *screen_type* argument is the same as described in the second form.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

For more information about preparing rendered image data for use with the HAME, please see Section 5.9.

10.4.11 HARLEQUIN

The HARLEQUIN saver lets you display images on Amiga Centre Scotland's Harlequin display board. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "HARLEQUIN"**
SAVE *filename type saver_opts*

The *type* argument must be either **"RAW"** or **"IMAGE"**.

The ARexx support for ACS's Harlequin is very similar to the support for the Impulse FireCracker 24. Each call to the HARLEQUIN saver from ARexx defines a stream of commands (defined as *saver_opts*) which are parsed and executed from left to right. These commands are:

ALPHA (ON | OFF)
This command turns the alpha channel of the currently selected screen on or off. If the currently selected board is not equipped with alpha channel support, an error will be returned.

BOARD_NO *num*
If you have more than one Harlequin installed, you can use this command to select which board subsequent commands will be sent to. All commands described here operate on the currently selected board.

CENTER
This command sets the display offsets to center the image currently in Amiga memory.

CLEAR
This command clears the currently selected screen.

DISPLAY (0 | 1)
This command is used to select which screen on the currently selected board will be displayed. If the currently selected board does not support double buffering, an error will be returned if you reference a screen which is not available.

GENLOCK (ON | OFF)
This command turns the genlock of the currently selected board on or off as specified. If the currently selected board supports double buffering, setting the genlock status for one screen affects the other screen as well.

IMAGE

This command causes the image currently in Amiga memory to be downloaded to the currently selected board and buffer at the current offsets.

IMAGE_TO (0 | 1)

This command is used to select which screen on the currently selected board will be the destination of all image data sent from the Amiga. If the currently selected board does not support double buffering, referencing the second buffer will generate an error.

LACE (ON | OFF)

This command turns the interlace mode of the currently selected screen on or off as specified. If the currently selected board supports double buffering, setting the interlace status of one screen does not affect the other.

SCREEN (ON | OFF)

This command turns the currently selected screen on or off as specified.

SET_DOX *left_edge*

This command specifies which column (**DOX** means "Destination Offset X") the image will be written to.

SET_DOY *top_edge*

This command specifies which row (**DOY** means "Destination Offset Y") the image will be written to.

SET_DWX *width*

This command sets the width of the image area to be written to the specified value. The required argument *width* is in Harlequin pixels.

SET_DWY *height*

This command sets the height of the image area to be written to the specified value. The required argument *height* is in Harlequin pixels.

SET_SOX *left_edge*

This command specifies the left edge of the image area *within the image* in Amiga memory which will be written to the board (**SOX** means "Source Offset X").

SET_SOY *top_edge*

This command specifies the top edge of the image area *within the image* in Amiga memory which will be written to the board (**SOY** means "Source Offset Y").

WIDTH (740 | 832 | 910)

This command sets display width of the currently selected board to the specified value. This command changes the width of both buffers if the currently selected board supports double buffering.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Since ARexx command lines invoking the Harlequin saver can be complex, finding an error on the command line can become difficult. The saver, therefore, returns in **ADPRO_RESULT** the number of the argument in which an error was found. The argument numbering begins with 1, which corresponds to the *type* field of the ARexx **SAVE** command. If an error is encountered, **RC** will return non-zero.

For instance, the following returns a 1 in **ADPRO_RESULT**:

```
SAVE "XXX" "NEITHER_RAW_NOR_IMAGE" "IMAGE"
```

If you request a data type (either **RAW** or **IMAGE**) which is not available, **RC** will be set to 10 and **ADPRO_RESULT** will be set to 1.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	1

10.4.12 IFF

The IFF saver can be controlled through a control panel or its ARexx interface.

ARexx Interface

Syntax: **SFORMAT "IFF"**
SAVE *image_filename type*

The IFF saver may be instructed to save **RAW**, **IMAGE**, or **SCREEN** data *type*.

The IFF saver does not have any *saver_opts*.

10.4.13 IMPULSE

The IMPULSE saver lets you export images in RGBN or RGB8 format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "IMPULSE"**
SAVE *filename* **"RAW"** *format*

The *format* argument must be either **"RGBN"** or **"RGB8"**.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.14 IV24

The IV24 saver lets you display images on Great Valley Products' IV24 display board. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "IV24"**
SAVE *filename type saver_opts*

The *type* argument must be **"RAW"** only.

The ARexx support for GVP's IV24 display board constitutes a language in-and-of-itself. Each call to the IV24 saver from ARexx defines a stream of commands (defined as *saver_opts*) which are parsed and executed from left to right. These commands are:

CENTER
This command sets the display offsets to center the image currently in Amiga memory.

IMAGE *n*
This command causes the image currently in Amiga memory to be downloaded to the IV24. The resulting IV24 display is shown for *n* ticks (fiftieths of a second). If *n* is 0, then the IV24 display is shown until the user clicks the left mouse button. A value for *n* must be supplied.

SET_DOX *left_edge*
This command specifies which column (**DOX** means "Destination Offset X") the image will be written to.

SET_DOY *top_edge*
This command specifies which row (**DOY** means "Destination Offset Y") the image will be written to.

SET_DWX *width*
This command sets the width of the image area to be written to the specified value. The required argument *width* is in IV24 pixels.

SET_DWY *height*
This command sets the height of the image area to be written to the specified value. The required argument *height* is in IV24 pixels.

SET_SOX *left_edge*

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

This command specifies the left edge of the image area *within the image* in Amiga memory which will be written to the board (**SOX** means "Source Offset X").

SET_SOY *top_edge*
This command specifies the top edge of the image area *within the image* in Amiga memory which will be written to the board (**SOY** means "Source Offset Y").

SCREEN_TYPE *stype*
The number specified by *stype* is used to determine what type of screen to construct. This number can be assembled using the values contained in Table 10.1. Note that the VGA, Super Hi-Res, Super72, and Default modes are not allowed.

Since ARexx command lines invoking the IV24 saver can be complex, finding an error on the command line can become difficult. The saver, therefore, returns in **ADPRO_RESULT** the number of the argument in which an error was found. The argument numbering begins with 1, which corresponds to the *type* argument of the ARexx **SAVE** command. If an error is encountered, **RC** will return non-zero.

For instance, the following returns a 1 in **ADPRO_RESULT**:

```
SAVE "XXX" "NOT_RAW" "CENTER" "IMAGE"
```

If raw image data is not available, **RC** will be set to 10 and **ADPRO_RESULT** will be set to 1.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	1

10.4.15 JPEG

The JPEG saver lets you compress images in JPEG (JFIF) format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "JPEG"**
SAVE *filename* **"RAW"** (1)
SAVE *filename* **"RAW"** (*quality* | **"BOOST"**) (2)

The JPEG saver may be instructed to save **RAW** data only; this keyword should be used as the value for the type argument (as shown above).

Mode	Value
Hi Resolution On	1
Interlace On	2
Horizontal Overscan On	8
Vertical Overscan On	16

Table 10.4: **SCREEN_TYPE** mask values for the OPALVISION saver.

If the first form is used, a default quality of 32 (without boost) is used.

If the second form is used, the *quality* argument (an integer between 1 and 100) will be used as the quality level for the JPEG encryption. If the keyword **"BOOST"** is present, the quality is modified to the boosted quality levels.

In the following example, a file called **"test.jpg"** will be created with quality 32 (no boost):

```
SAVE "test.jpg" "RAW" 32
```

The following examples are the same. A file called "test.jpg" will be created with quality 100 (with boost):

```
SAVE "test.jpg" "RAW" "BOOST" 100
SAVE "test.jpg" "RAW" 100 "BOOST"
```

If load was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.16 OPALVISION

The OPALVISION saver lets you display images on Opal Technology's / Centaur Development's OpalVision display board. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "OPALVISION"**
SAVE "XXX" *type saver_opts*

The *type* argument must be one of **"RAW"**, **"IMAGE"**, or **"SCREEN"**.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The *saver_opts* for the OPALVISION saver are:

SCREEN_TYPE *stype*
: The display resolution, similar to ADPro's **Screen Controls** ARexx interface. Note, however, that only the flags specified in Table 10.4 are allowed.

GUI Equivalent: Resolution control rocker switch.
Constraints: See table above.
Required: No

DURATION *t*
: Amount of time in ticks (fiftieths of a second) to display the image.

GUI Equivalent: None
Constraints: $0 \leq t$
Required: No

A couple of examples are:

```
SAVE "XXX" "RAW" SCREEN_TYPE 3
```

This displays Raw image data in hires-interlaced mode.

```
SAVE "XXX" "IMAGE" SCREEN_TYPE 8 DURATION 100
```

This displays Rendered image data in lores-overscan for 2 seconds.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error string

10.4.17 PCX

The PCX saver lets you export images in PCX format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "PCX"**
SAVE *filename type pcx_type*

The *type* argument must be either **"RAW"** or **"IMAGE"**.

PCX files can come in four flavors: CGA, EGA, VGA, and 24 bit.

If you specify that you wish to save **RAW** data, any *saver_opts* which you might have included will be ignored. The resulting PCX file will either be of the 24 bit-plane color variety (if you have color image data loaded at the time) or will be a 256 shade VGA file (if you have grayscale image data loaded at the time).

If you wish to save **IMAGE** data, a *pcx_type* chosen from one of **CGA**, **EGA**, or **VGA** must be specified, unless the rendered image data available at the time is unambiguously VGA (more than 16 colors).

If you attempt to save an image rendered in a uniquely Amiga mode, an error is returned.

For example, if 256 color rendered image data is available, either of the following two lines are acceptable (to save a VGA file).

```
SAVE "test.pcx" "IMAGE"
SAVE "test.pcx" "IMAGE" "VGA"
```

And both of the following two lines would produce errors (under the same circumstances).

```
SAVE "test.pcx" "IMAGE" "CGA"
SAVE "test.pcx" "IMAGE" "EGA"
```

If 16 color rendered image data were available, then the following line would return an error since you can have 16 color VGA and 16 color EGA files.

```
SAVE "test.pcx" "IMAGE"
```

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.18 POSTSCRIPT

The POSTSCRIPT saver lets you export images in PostScript or Encapsulated PostScript format. It can be accessed only through the main control screen. No ARexx control exists at this time.

10.4.19 PREFPRINTER

The PREFPRINTER saver lets you print images to any printer that has a Preferences-supported printer driver that supports strip printing. It can be accessed through the main control screen and through ARexx.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

ARexx Interface

Syntax: `SFORMAT "PREFPRINTER"`
`SAVE "XXX"` *type saver_opts*

The *type* argument must be `"RAW"` only.

The PREFPRINTER saver can be called from ARexx by specifying as many arguments (*saver_opts*) as you need, as long as you conform to the following order:

orientation
Which must be either `"HORIZONTAL"` or `"VERTICAL"`.

dither_mode [*mask_size*]
Which must be one of `"FLOYD"`, `"ORDERED"`, `"DHORIZONTAL"`, `"DVERTICAL"`, `"FWDBRICK"`, `"BCKBRICK"`, `"FWDDIAGONAL"`, `"BCKDIAGONAL"`, `"HALFTONEA"`, or `"HALFTONEB"`.

If *dither_mode* is set to anything other than `"FLOYD"` or `ASCII`, then you must specify the dither *mask_size* (one of 2, 4, 8, or 16).

density
Which ranges from 0 to 7. If set to 0, then the density is taken from its last used value. If *density* is from 1 to 7, it specifies the Preferences density of the printer being used.

print_width
Which corresponds to the **Print Dimensions Width** value.

print_height
Which corresponds to the **Print Dimensions Height** value.

print_left
Which corresponds to the **Print Dimensions Left** value.

print_top
Which corresponds to the **Print Dimensions Top** value.

page_width
Which corresponds to the **Print Definitions Width** value.

page_height
Which corresponds to the **Print Definitions Height** value.

page_left_margin
Which corresponds to the **Print Definitions Left** value.

page_top_margin
Which corresponds to the **Print Definitions Top** value.

paper_height
Which corresponds to the **Paper Dimensions Height** value.

For example, to use all default values you would say:

```
SAVE "XXX" "RAW"
```

To specify vertical printing and Floyd-Steinberg dithering, you would say:

```
SAVE "XXX" "RAW" "VERTICAL" "FLOYD"
```

To specify density 3 with horizontal printing and ordered dithering (with a mask size of 16), you would say:

```
SAVE "XXX" "RAW" "HORIZONTAL" "ORDERED" 16 3
```

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.20 QRT

The QRT saver lets you export images in QRT format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "QRT"**
SAVE *filename type*

The *type* argument must be **"RAW"** only.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.21 RESOLVER

The RESOLVER saver lets you send images to the DMI Resolver display board. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "RESOLVER"**
SAVE "XXX" *type saver_opts*

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The *type* argument must be **"IMAGE"** only, or **"RAW"** for 8-bit grayscale images.

The ARexx support for the DMI Resolver saver constitutes a language in-and-of-itself. From ARexx, each call to the DMI Resolver saver defines a stream of commands (defined as *saver_opts*) which are parsed and executed from left to right. These commands are:

BOARD_MODE *num*
: This command sets the display size based on the currently defined size numbers. These mode numbers are defined within the DMI configuration software.

  If an invalid number is specified, an error will occur and the saver will exit with an error code of 10.

BOARD_NO *num*
: If you have more than one DMI Resolver installed, this command selects to which board subsequent commands will be sent. All commands described here operate on the currently selected board.

CENTER
: This command sets the display offsets to center the image currently in Amiga memory.

CLEAR
: This command clears the currently selected board.

IMAGE
: This command causes the image currently in Amiga memory to be downloaded to the currently selected board at the current offsets.

SET_DOX *left_edge*
: This command specifies which column (**DOX** means "Destination Offset X") the image will be written to.

SET_DOY *top_edge*
: This command specifies which row (**DOY** means "Destination Offset Y") the image will be written to.

SET_DWX *width*
: This command sets the width of the image area to be written to the specified value. The required argument *width* is in DMI Resolver pixels.

SET_DWY *height*
: This command sets the height of the image area to be written to the specified value. The required argument *height* is in DMI Resolver pixels.

SET_SOX *left_edge*
: This command specifies the left edge of the image area within the image in memory which will be written to the board (**SOX** means "Source Offset X").

SET_SOY *top_edge*

This command specifies the top edge of the image area within the image in memory which will be written to the board (**SOY** means "Source Offset Y").

Since ARexx command lines invoking the DMI Resolver saver can be complex, finding an error on the command line can become difficult. The saver, therefore, returns in **ADPRO_RESULT** the number of the argument in which an error was found. The argument numbering begins with 1, which corresponds to the *type* argument of the ARexx **SAVE** command. If an error is encountered, **RC** will return non-zero.

For instance, the following returns a 1 in **ADPRO_RESULT**:

```
SAVE "XXX" "NEITHER_RAW_NOR_RENDERED" "IMAGE"
```

Examples of normal usage:

```
SAVE "XXX" "IMAGE" "CLEAR" "CENTER" "IMAGE"
```

This will clear the board, center, and download the image to the current selected DMI Resolver.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.4.22 SCULPT

The SCULPT saver lets you export images in SCULPT format. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "SCULPT"**
SAVE *filename type*

The *type* argument must be **"RAW"** only.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.4.23 SEPARATE

The SEPARATE saver lets you create color separations. It can be accessed through the main control screen and through ARexx.

ARexx Interface

Syntax: **SFORMAT "SEPARATE"**
SEPARATE_opts
SAVE *rootname plane*

This section describes how the commands relating to color separation are used from ARexx. Note that all image data saved by ADPro's color separation routines are saved in 4 or 8 bit-plane IFF files. If you require saved color separation data in a different format, use ADPro to convert the IFF format file into the format you need.

Before you can do a separation (with the **SAVE** command), you should set up your options with zero or more invocations of the following commands:

UCR (1)
UCR *value* (2)

The first form of the command returns the current UCR (Under Color Removal) setting in **ADPRO_RESULT**.

The second form of the command allows the UCR value to be set to the supplied value which must be in the range of 0 to 100.

If the supplied value is outside the range of 0 to 100, then the **RC** variable will be set to 10 and **ADPRO_RESULT** will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, **RC** returns as 0 and **ADPRO_RESULT** returns with the *previous* setting.

GCR (1)
GCR *value* (2)

The first form of the command returns the current GCR (Gray Component Replacement) setting in **ADPRO_RESULT**.

The second form of the command allows the GCR value to be set to the supplied value which must be in the range of 0 to 100.

If the supplied value is outside the range of 0 to 100, then the **RC** variable will be set to 10 and **ADPRO_RESULT** will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, **RC** returns as 0 and **ADPRO_RESULT** returns with the *previous* setting.

SEPARATION_MAP (1)
SEPARATION_MAP *value* (2)

The first form of the command returns the current **CMAP** rocker switch setting in **ADPRO_RESULT**. This will be in the form of text reading either **COLOR** or **GRAY**.

The second form allows you to set the value of the **CMAP** rocker switch. The single supplied argument must be either COLOR or GRAY. If the call was successful, the value returned in ADPRO_RESULT is the previous **CMAP** rocker switch setting.

If the call is unsuccessful, the value returned in RC is 10. A value of 0 is returned in RC if the call (either form) completed properly.

MAGENTA_CORRECTION (1)

MAGENTA_CORRECTION *value* (2)

The first form of the command returns the current magenta correction setting in ADPRO_RESULT.

The second form of the command allows the magenta correction value to be set to the supplied value which must be in the range of 0 to 100.

If the supplied value is outside the range of 0 to 100, then the RC variable will be set to 10 and ADPRO_RESULT will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, RC returns as 0 and ADPRO_RESULT returns with the *previous* setting.

YELLOW_CORRECTION (1)

YELLOW_CORRECTION *value* (2)

The first form of the command returns the current yellow correction setting in ADPRO_RESULT.

The second form of the command allows the yellow correction value to be set to the supplied value which must be in the range of 0 to 100.

If the supplied value is outside the range of 0 to 100, then the RC variable will be set to 10 and ADPRO_RESULT will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, RC returns as 0 and ADPRO_RESULT returns with the *previous* setting.

INK_CORRECTION (1)

INK_CORRECTION (0 | 1) (2)

The first form of the command returns the current state of the ink correction setting in ADPRO_RESULT.

The second form of the command allows the state of the ink correction setting to be set to the specified state which must either be 0 or 1. A value of 0 disables ink correction. A value of 1 enables ink correction.

If the supplied value is neither 0 nor 1 then the RC variable will be set to 10 and ADPRO_RESULT will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, RC returns as 0 and ADPRO_RESULT returns with the *previous* state of the ink correction setting.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

SEPARATION_DEPTH (1)

SEPARATION_DEPTH (4 | 8) (2)

The first form of the command returns the current depth setting for color separations. The returned value will be either a 4 or 8 and is returned in **ADPRO_RESULT**.

The second form of the command allows the depth of color separations to be set to the supplied value which must be 4 or 8.

If the supplied value is neither 4 nor 8, then the **RC** variable will be set to 10 and the **ADPRO_RESULT** will be set to the string **Invalid Argument To Function.**

If the supplied value is accepted, **RC** returns as 0 and **ADPRO_RESULT** returns with the *previous* color separation depth setting.

SEPARATION_TYPE (1)

SEPARATION_TYPE (RGB | CMY | CMYK) (2)

The first form of the command returns the current separation type. This will be returned in **ADPRO_RESULT** and will be one of the following strings: **RGB**, **CMY**, or **CMYK**.

The second form of the command sets the separation type to the supplied value which must match one of the above mentioned strings.

If the supplied string is not acceptable, then the **RC** variable will be set to 10 and the **ADPRO_RESULT** will be set to the string **Invalid Argument To Function.** If the supplied string is accepted, **RC** is returned as zero, and **ADPRO_RESULT** will contain the *previous* separation type.

USE_SEPARATION_DEFAULTS

This command sets the **UCR**, **GCR**, and ink corrections values to their default settings. This command never returns an error.

SEPARATE *rootname plane*

This command is equivalent to the actual **SAVE** *rootname plane* calling sequence. It has been retained for compatibility with current ARexx scripts.

The actual **SAVE** command causes a separation to actually take place. If the separation type is **RGB**, then the *plane* argument may be one or all of r, g, and b. If the separation type is **CMY**, then the *plane* argument may be one or all of c, m, and y. If the separation type is **CMYK**, then the *plane* argument may be one or all of c, m, y, or k.

rootname is the name to be used for the separation output *without any file name extensions appended.* ADPro will append the extensions appropriate for the specific type of separation data that it writes.

Do not specify an extension in the supplied file name as ADPro will supply an appropriate extension automatically.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	non-zero value	undefined

10.4.24 TEMP

The TEMP saver lets you copy the raw image data in the primary image buffer to the temporary image buffer. It can be controlled through the main ADPro control screen or through its ARexx interface.

ARexx Interface

Syntax: **SFORMAT "TEMP"**
SAVE "XXX" "RAW"

The TEMP saver save command places the raw image data into the temporary image buffer.

The TEMP saver does not have any *saver_opts*.

If load was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5 Operators

All of the operators can be controlled through an operator-specific control panel (or screen of panels) or through their ARexx interfaces.

10.5.1 General Information

ARexx Interface

Operator Format

Syntax: **OFORMAT** (1)
OFORMAT *format* (2)

The first form of the command will fill the **ADPRO_RESULT** field with the name of the currently selected Operator Format, if results are requested.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The second form of the command sets the Operator Format to the specified string.

If module	RC	ADPRO_RESULT
exists	0	the *previously* selected Operator Format
does not exist	10	"Unknown Module"

Calling an Operator

Syntax: **OPERATE** *operator_args*
This command invokes the currently selected operator (selected with the **OFORMAT** command or set by the last **OPERATOR** invocation) using the given arguments. This command is equivalent to the **Execute Op** button on the control panel.

Syntax: **OPERATOR** *operator_name* [*operator_args*]
The **OPERATOR** command is a general purpose front-end to run-time loadable ADPro operators. Operators are independent programs typically performing an image processing function which modifies the raw and/or rendered image data currently available.

As given above, the generalized syntax of the **OPERATOR** command requires the operator name to be given as the command's first argument. Additional arguments may be required depending upon the particular operator being called.

The following sections detail the operators provided as standard with ADPro.

10.5.2 Apply_Map

The Apply_Map operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "APPLY_MAP"**
This operator causes any raw image data to be replaced by the raw image data after filtering through the current color controls.

If raw data is	RC	ADPRO_RESULT
available	0	undefined
not available	10	undefined

10.5.3 Blur

The Blur operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "BLUR"` *cw th* (1)
`OPERATOR "BLUR"` *cw th* `"TEST"` (2)

The first form of the operator will blur any raw data present using the supplied center weight (*cw*) and threshold (*th*).

The second form of the operator will test the supplied values by performing the appropriate calculations but not actually modify the image.

Both forms return the number of pixels which would be (in the second form) or were (as in the first form) affected by the operation in `ADPRO_RESULT`.

The *cw* value may range from 0 to 16; *th* may range from 0 to 255.

If	RC	ADPRO_RESULT
operation was successful	0	undefined
raw data is not available or values are out of range	10	undefined

10.5.4 Broadcast_Limit

The Broadcast_Limit operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "BROADCAST_LIMIT"` (1)
`OPERATOR "BROADCAST_LIMIT"` *s* (2)
`OPERATOR "BROADCAST_LIMIT"` *s lt* (3)
`OPERATOR "BROADCAST_LIMIT"` *s lt other_opts* (4)

The first form will apply the Broadcast_Limit operator to any raw data present using the settings that were previously defined. Optional arguments may be specified to change the settings of the operator.

This operator returns the number of pixels affected by the operation in `ADPRO_RESULT`.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The second form allows you to specify (with *s*) the television standard which will be used in the calculations performed. The *s* should be replaced by either the string "NTSC" or "PAL".

The third form allows you to specify (with *lt*) the limit type. The limit type may be specified as the string "LUMINANCE" or "SATURATION". This has the same effect as toggling the **Luminance/Saturation** rocker switch on the Broadcast_Limit control panel.

The fourth form allows you to specify additional options which can appear in any order. These include:

"COMPOS_LIM" *c*
: This allows you to change the maximum value allowed for the composite signal that would be generated if the image was to be displayed on a NTSC or PAL television. The *c* value has a range from 0.00 to 199.99. If the value given is out of the acceptable range, ADPro will return a result of 10 in the RC variable. The ADPRO_RESULT variable will contain the argument number (beginning with 1 corresponding to the *s* parameter) which caused the error.

"CHROMA_LIM" *cv*
: This allows you to change the maximum value allowed for the chroma in the image. The *cv* value has a range from 0.00 to 99.99. If the given value is out of the acceptable range, ADPro will return a result of 10 in the RC variable. The ADPRO_RESULT variable will contain the argument number which caused the error (beginning with 1 corresponding to the *s* parameter).

TEST
: Specifying this argument allows you to "pencil test" the Broadcast_Limit operation. This allows you to test your settings of the operator, without changing any of the raw data. The number of pixels which would have been affected will be returned in the ADPRO_RESULT variable.

DEFAULT_LIM
: Specifying this command causes the composite and chroma values to be reset to the default settings, which are 110.0 for composite, and 50.0 for the chroma setting.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error argument number

10.5.5 Collapse

The Collapse operator can be controlled through its control panel (that opens on its own screen) or through its ARexx interface.

Menu Bar

Project Menu	
About...	**Rt-Amiga ?**
Accept	**Rt-Amiga Y**
Quit	**Rt-Amiga Q**

ARexx Interface

Syntax: **OPERATOR "COLLAPSE"** *arguments*
where *arguments* can be one or more of the following (specified in any order):

AMOUNT *a*
Set the degree of collapse.

GUI Equivalent: **Amount (Deg)** value.
Constraints: $0 \leq a \leq 100.0$
Required: Yes

BLUR_RADIUS *b*
Set the percent of the radius (starting at the outer part of the circle) that will be blurred.

GUI Equivalent: **BRadius(0-99)** value.
Constraints: $0 \leq b \leq 99$
Required: Yes

CENTER *xpos ypos*
Set the location of the collapse circle's center.

GUI Equivalent: **Center X (Px)** and **Center Y (Px)** values.
Constraints: $-30000 \leq xpos \leq 30000$; $-30000 \leq ypos \leq 30000$
Required: Yes

RADIUS *r*
Set the radius of the collapse circle.

GUI Equivalent: **Radius (Px)** value.
Constraints: $5 \leq r \leq 15000$
Required: Yes

(QUALITY_FAST | QUALITY_HIGH)
Set the precision of the collapse operation.

GUI Equivalent: **Quality** rocker switch setting.
Constraints: None
Required: Yes

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error message

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.5.6 Color_To_Gray

The Color_To_Gray operator uses a control panel to specify how to do the conversion. This operator also has an ARexx interface.

Control Panel

Keyboard Equivalents	
Perform the Operation	`Shift + Return`

ARexx Interface

Syntax: `OPERATOR "COLOR_TO_GRAY"` (1)
`OPERATOR "COLOR_TO_GRAY"` *rw gw bw* (2)

This operator converts raw color image data into raw grayscale image data.

This operator requires raw color data.

The first form of the command converts color information into grayscale using the weighting factors used by the NTSC standard.

The second form allows you to specify the weighting to be used for the red, green, and blue components of the image. Weights are scaled to a range of 0 for a zero weight to 10000 for a 100 percent weight. Weights may be negative or exceed 100 percent.

If raw color data is	RC	ADPRO_RESULT
available	0	undefined
not available	10	"Error"

10.5.7 Colorize

The Colorize operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax:
`OPERATOR "COLORIZE"` *t* (1)
`OPERATOR "COLORIZE"` *t pf pt* (2)
`OPERATOR "COLORIZE"` *t pf pt f* (3)
`OPERATOR "COLORIZE"` *t pf pt f r g b* (4)
`OPERATOR "COLORIZE"` *t pf pt f r g b htp* (5)
`OPERATOR "COLORIZE"` *t pf pt f r g b htp hf ht* (6)

This operator requires the presence of raw grayscale image data which it converts into raw color data where some or all of the gray image has actually been colored in.

The first form performs a colorization of the specified type (specified in *t* which must be one of **LOADED**, **RANDOM**, **FALSE**, or **HSV**). For an explanation of what function each type of colorization will perform, please turn to Section 6.6. In this form, the value of all other arguments are taken to their most recently set values.

The second form allows you to specify a range of grayscales which will be affected by the colorizing operation. Note that this range always spreads numerically upwards from the *pf* value to the *pt* value (i.e., from and to) and that, if necessary, it will wrap back to 0 if it reaches 255.

The third form allows you to specify (with *f*) what is to be done with those shades of gray which lie outside the range you specify with *pf* and *pt*. If the *f* value is set to "-1" (note that the quotation marks are required when specifying a negative number), then the gray shades are passed through without change. If *f* is set to a value between 0 and 255 (inclusive), pixels outside the range specified with *pf* and *pt* are set to the gray shade specified by *f*.

The fourth form allows you to set the base R, G and B color used by the **HSV** and **RANDOM** colorizations.

The fifth form allows you to select which type of HSV colorization you wish to perform. The argument *htp* is valid only when *t* is set to **HSV**. This argument may be one of **VALUE**, **HUE**, or **SATURATION**.

The sixth form allows you to specify the range of hues, values, or saturations which will be used during an HSV colorization. The *hf* and *ht* arguments are valid only when *t* is set to **HSV**. The *hf* and *ht* range depends upon the type of HSV colorization being specified (in the *htp* variable).

hf
: If *htp* is "**VALUE**", this may be between 0 and 1.
If *htp* is "**SATURATION**", this may be between 0 and 1.
If *htp* is "**HUE**", this may be between 0 and 10000.

ht
: If *htp* is "**VALUE**", this may be between 0 and 1.
If *htp* is "**SATURATION**", this may be between 0 and 1.
If *htp* is "**HUE**", this may be between 0 and 10000.

This operator will set **RC** to non-zero if an error has occurred or will return 0 in **RC** if the operation was successful.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

10.5.8 Convolve

The Convolve operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "CONVOLVE"** *matrix_file mix thresh*

This operator causes the convolution matrix contained in the file named *matrix_file* to be passed over the current image data using the value of *mix* as the mixing setting and the value of *thresh* as the threshold setting.

NOTE The string you pick for *matrix_file* must be a full path name to the matrix file you wish to use. For example, you might specify **"ADPRO:Convolutions/MatrixName"**.

The valid range of *mix* settings is 1 to 100. The valid range of *thresh* is from 0 to 255.

If the operation is aborted by the user, **RC** will return as 10 and **ADPRO_RESULT** will contain the string **Aborted**. If the operation succeeded, **RC** will return as a value of 0 and **ADPRO_RESULT** will contain the name of the operator which was used previous to this one.

If operation was	RC	ADPRO_RESULT
successful	0	*previousmatrixfile*
aborted	10	**Aborted**

10.5.9 Crop_Image

The Crop_Image operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "CROP_IMAGE"** *width height* (1)
OPERATOR "CROP_IMAGE" *width height left top* (2)

This operator can be used to select and preserve a rectangular region within an image and discard all data outside the selected region.

The first form of the command preserves the specified *width* and *height* starting at the upper left-hand corner of the original image.

The second form allows you to specify an offset (*left* and *top*) relative to the top left-hand corner of the original image.

Note that the offsets (if specified) cannot be negative nor can the offsets plus the width or height exceed the width or height of the original image. If the supplied parameters are not accepted, then the **RC** value will be set to 10. If the operation is successful, **RC** is returned with a value of 0.

From ARexx, you should use this operator rather than Crop_Visual.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.10 Crop_Visual

The Crop_Visual operator can be controlled through a control panel on its own control screen, as well as through its ARexx interface.

Control Panel

Keyboard Equivalents	
Zooming In	**z**
Zip/Unzip the Control Panel	**u**
Perform the Operation	**Shift + Return**

ARexx Interface

Syntax: **OPERATOR "CROP_VISUAL"** *width height* (1)
OPERATOR "CROP_VISUAL" *width height left top* (2)

This operator can be used to select and preserve a rectangular region within an image and discard all data outside the selected region.

The first form lets you specify the width and height of the cropped area using an offset of 0,0 (the upper left corner of the original image).

The second form lets you specify the width, height, and left and top offsets.

Note that if you specify offsets (second form), they cannot be negative nor can the offsets plus the width or height exceed the width or height of the original image.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.11 DCTV

The DCTV operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "DCTV"`

This operator converts the currently loaded raw image data into rendered data usable by the DCTV display device. You can use the IFF saver to save the converted data to disk or you can use the **ADPRO_DISPLAY** command to view the DCTV data from within ARexx.

The DCTV operator takes its display size and type (i.e., interlace on or not, PAL or NTSC, etc.) from ADPro's current screen control settings.

If ADPro is currently set to render 8 colors, then a three bit-plane DCTV image will be created. If ADPro is set to render any other type of image (not 8 colors), then a four bit-plane DCTV image will be created.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.12 Define_Pxl_Aspect

The Define_Pxl_Aspect operator can be controlled through its control panel or its ARexx interface.

Control Panel

Keyboard Equivalents	
Toggle Between Input Fields	`Alt + Return`
Perform the Operation	`Shift + Return`

ARexx Interface

Syntax: **OPERATOR "DEFINE_PXL_ASPECT"** *arguments*

where *arguments* can be zero or more of the following (listed in any order):

XASPECT *xasp*
Set the horizontal pixel aspect.

GUI Equivalent: **New (X) Aspect** value.
Constraints: $1 \leq xasp \leq 240$
Required: No

YASPECT *yasp*
Set the vertical pixel aspect.

GUI Equivalent: **New (Y) Aspect** value.
Constraints: $1 \leq yasp \leq 240$
Required: No

XRES *xres*
Set the horizontal resolution (DPI).

GUI Equivalent: **X DPI** value.
Constraints: $1 \leq xres \leq 9999$
Required: No

YRES *yres*
Set the vertical resolution (DPI).

GUI Equivalent: **Y DPI** value.
Constraints: $1 \leq yres \leq 9999$
Required: No

This operator allows you to define the current image's pixel aspect and resolution information. If no arguments are given, then the current aspect and resolution information will be returned in a string (**ADPRO_RESULT**) of the following form:

xasp yasp xres yres width height

where *xasp* and *yasp* are the x and y aspect of the pixels in the currently defined image. The *xres* and *yres* values are the currently defined x and y resolutions. The *width* and *height* values are the width and height of the image in pixels. You can determine the size (in inches) that this image is asking to be by dividing the width or height values by the *xres* or *yres* fields, respectively.

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If	RC	ADPRO_RESULT
no values are given	0	the *current* pixel aspect and resolution information
values are correct	0	the *previous* pixel aspect and resolution information
any of the values are out of range or do not make sense	non-zero	undefined

10.5.13 DeInterlace

The DeInterlace operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "DEINTERLACE"`

This operator splits the currently defined image into two images, stacked one above the other. In the top half, the even numbered scanlines of the former image can be found. In the bottom half, what were formerly the odd numbered scanlines can be found.

This operator is particularly useful for those video digitizing moving scenes.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.14 Displace_Pixel

The Displace_Pixel operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax:
`OPERATOR "DISPLACE_PIXEL"` (1)
`OPERATOR "DISPLACE_PIXEL"` *rad* (2)
`OPERATOR "DISPLACE_PIXEL"` *rad prob* (3)
`OPERATOR "DISPLACE_PIXEL"` *rad prob seed* (4)

The first form performs the operation using the previously-defined settings.

The second form takes a value for the displacement circle's radius (*rad*), but uses the other attributes' current values. The *rad* value can be between 1 and half of the image's width.

The third form takes the *rad* and probability (*prob*) values, but uses the other attributes' current values. The *prob* value can be between 1 and 100.

The fourth form requires that you specify all of the attributes' values, *rad*, *prob*, and the randomizer's *seed* value. This can range between -99999999 and 99999999.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	the *previous* operator

10.5.15 Dynamic_Range

The Dynamic_Range operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "DYNAMIC_RANGE"` (1)
`OPERATOR "DYNAMIC_RANGE"` *newmin newmax* (2)

The first form of this operator examines any currently loaded raw image data and returns the minimum and maximum values it finds in `ADPRO_RESULT`. It does not change the data in any way.

The second form of the command remaps any currently loaded raw image data to lie between the supplied new minimum and maximum. These values must be between 0 and 255 inclusively. The old minimum and maximum is returned in `ADPRO_RESULT`.

In both forms, when a maximum and minimum are returned in `ADPRO_RESULT`, they are returned in the following format:

min max

If the operator executed properly, the `RC` variable is set to 0. If an error is found, the `RC` variable is set to 10 and the `ADPRO_RESULT` variable is not set.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.16 Gray_To_Color

The Gray_To_Color operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "GRAY_TO_COLOR"`
This operator converts grayscale raw image data into raw color image data. The resulting raw image has identical red, green and blue components.

If	RC	ADPRO_RESULT
successful	0	undefined
raw grayscale image data is not present when this operator is executed or there is not enough memory available to perform the conversion	10	undefined

10.5.17 Halve

The Halve operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "HALVE"`
This operator scales both the width and height of the currently defined image to one half their original sizes. It does so much more quickly than ADPro's general purpose scaling function but with the same precision.

This operator requires raw data.

If	RC	ADPRO_RESULT
operation was successful	0	undefined
operation wasn't successful (no raw data available)	10	"Error"

10.5.18 Horizontal_Flip

The Horizontal_Flip operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "HORIZONTAL_FLIP"`

This operator flips the available raw data horizontally about its center.

This operator requires raw data.

If	RC	ADPRO_RESULT
successful	0	undefined
no raw data is available	10	"Error"

10.5.19 Intensity_Range

The Intensity_Range operator can be controlled through its control panel (that opens on ADPro's screen). It does not have its own dedicated ARexx interface for settings values, but it can be initiated through ARexx.

ARexx Interface

Syntax: `OPERATOR "INTENSITY_RANGE"`

If operation was	RC	ADPRO_RESULT
successful	0	the *previous* operator
not successful	10	Error

10.5.20 Interlace

The Interlace operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "INTERLACE"`

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

This operator performs an interlacing function by assuming that the currently loaded image actually contains two separate images (of the same height) stacked one upon the other.

It produces raw image data which contains scanlines chosen alternately from the top and bottom halves of the previously loaded raw image data.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.21 KillTemp

The KillTemp operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "KILLTEMP"`

This operator causes any image data currently in the temporary buffer to be erased.

If raw data is	RC	ADPRO_RESULT
available	0	undefined
not available	10	Error message

10.5.22 Line_Art

The Line_Art operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "LINE_ART"`

This operator causes a proprietary edge detection algorithm to be passed over the current raw grayscale image data. The resulting image can sometimes be very good quality line art.

This operator requires raw grayscale data. If no raw grayscale data is available, `RC` will be set to 10 and `ADPRO_RESULT` will be set to `Error`. This operator sets `RC` to a value of 0 to signify that no error occurred.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error

10.5.23 Median_Filter

The Median_Filter operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "MEDIAN_FILTER"` *th* (1)
`OPERATOR "MEDIAN_FILTER"` *th* `"TEST"` (2)

The first form will apply the Median_Filter operator to any raw data present using the supplied threshold (*th*).

The second form of the operator will test the supplied threshold by performing the appropriate calculations but not actually modify the image.

Both forms return the number of pixels which would be (in the second form) or were (as in the first form) affected by the operation in `ADPRO_RESULT`.

th may range from 0 to 255. If *th* lies outside the specified range, or there is no raw data available, a value of 10 will be placed in `RC`. Otherwise, `RC` returns set to 0.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.24 Negative

The Negative operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "NEGATIVE"`

This operator inverts whatever raw image data is available. If raw grayscale data is available, this operator produces a "black and white negative." If color raw data is available, then this operator produces a "color negative".

This operator requires raw data.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If	RC	ADPRO_RESULT
operation was successful	0	undefined
operation wasn't successful (no raw data available)	10	"Error"

10.5.25 Polar_Mosaic

The Polar_Mosaic operator can be controlled through its control panel (that opens on its own screen) or through its ARexx interface.

Menu Bar

Project Menu	
About...	Rt-Amiga ?
Accept	Rt-Amiga Y
Quit	Rt-Amiga Q

ARexx Interface

Syntax: `OPERATOR "POLAR_MOSAIC"` *arguments*

where *arguments* can be one or more of the following (specified in any order):

`TRACKS` *t*

Set the number of circular tracks to create.

GUI Equivalent: **Tracks** value.
Constraints: $1 \leq t \leq 10000$
Required: Yes

`BLUR_RADIUS` *b*

Set the percent of the radius (starting at the outer part of the circle) that will be blurred.

GUI Equivalent: **BRadius(0-99)** value.
Constraints: $0 \leq b \leq 99$
Required: Yes

`CENTER` *xpos ypos*

Set the location of the mosaic circle's center.

GUI Equivalent: **Center X (Px)** and **Center Y (Px)** values.
Constraints: $-30000 \leq xpos \leq 30000$; $-30000 \leq ypos \leq 30000$
Required: Yes

RADIUS *r*

Set the radius of the mosaic circle.

GUI Equivalent: **Radius (Px)** value.
Constraints: $5 \leq r \leq 15000$
Required: Yes

SLICES *s*

Set the number of "pie-style" divisions in the circle.

GUI Equivalent: **Slices** value.
Constraints: $1 \leq s \leq 10000$
Required: Yes

(QUALITY_FAST | QUALITY_HIGH)

Set the precision of the mosaic operation.

GUI Equivalent: **Quality** rocker switch setting.
Constraints: None
Required: Yes

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error message

10.5.26 Rectangle

The Rectangle operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax:
OPERATOR "RECTANGLE" *le te w h* (1)
OPERATOR "RECTANGLE" *le te w h thk* (2)
OPERATOR "RECTANGLE" *le te w h thk r g b* (3)
OPERATOR "RECTANGLE" *le te w h thk r g b mix* (4)

This operator draws a rectangle into the currently defined raw image. Required arguments include the left edge, top edge, width, and height of the rectangle. The default values for the other arguments (if not specified) are: thickness of 1, color of black, and a mix of 100 percent.

Specifying a thickness of "-1" indicates a filled rectangle. Note that you must include the -1 in quotation marks to prevent ARexx from thinking you wanted to subtract one from the previous argument.

Rectangle color must always be specified by an R, G, B triple. For drawing into a grayscale image, the red value is used.

The mix level must range between 1 and 100.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If any of the supplied arguments are invalid, of no raw image data is available to draw into, the RC variable is set to 10. Otherwise, it is set to 0 and the rectangle is drawn.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.27 Rectangle_Visual

The Rectangle_Visual operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "RECTANGLE_VISUAL"` *arguments*

This operator draws a rectangle into the currently defined raw image. However, we do not recommend the use of this operator from ARexx due to its size. This operator is best used from the user interface because of its visual nature. To draw rectangles from ARexx, use Rectangle instead.

If you insist on using this operator from ARexx, its programming information and behavior is exactly the same as Rectangle.

10.5.28 Rem_Isolated_Pxls

The Rem_Isolated_Pxls operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "REM_ISOLATED_PXLS"`

This operator examines every pixel in a set of rendered data comparing each pixel to its neighbors. If all of a given pixel's neighbors are the same color and that color is different from the given pixel, the pixel is changed to match the color of its neighbors.

This operator requires rendered data. If no rendered data is available, RC will be set to 10 and ADPRO_RESULT is undefined. This operator sets RC to a value of 0 to signify that no error occurred.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.29 Rendered_To_Raw

The Rendered_To_Raw operator does not bring up any control panel of its own. Once you select this operator, a meter window will appear and the operation will start immediately. It can also be started through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "RENDERED_TO_RAW"`

This operator converts the rendered image data currently loaded in ADPro's memory back into raw data. The type of the raw data produced (color or grayscale) depends upon which type of rendered image was converted.

If	RC	ADPRO_RESULT
successful	0	undefined
not successful (not enough memory for the raw data)	10	undefined

10.5.30 Roll

The Roll operator can be controlled through its control panel (shown atop ADPro's main screen), as well as through its ARexx interface.

ARexx Interface

Syntax: `OPERATOR "ROLL"` *direction pixels* (1)
`OPERATOR "ROLL"` *direction pixels* `("WRAP" | "NO_WRAP")` (2)

The first form will apply the Roll operator to any raw data present. The minimum requirements are to specify a roll *direction* and the number of *pixels* to roll the picture.

Substitute one of the following strings for *direction*: `"DOWN"`, `"LEFT"`, `"UP"`, `"RIGHT"`, `"UPLEFT"`, `"UPRIGHT"`, `"DOWNRIGHT"`, or `"DOWNLEFT"`.

The second form will allow you to specify whether or not the image will wrap as it rolls. If `"WRAP"` is specified, any portion of the image moved off the screen on one side, will appear in the vacated space on the other side. If `"NO_WRAP"` is specified, the vacated space is filled with black.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

If	RC	ADPRO_RESULT
operation was successful	0	undefined
operation wasn't successful (no raw data available)	10	"Error"

10.5.31 Rotate

The Rotate operator can be controlled through its control panel (shown on a separate control screen), as well as through its ARexx interface.

Menu Bar

Project Menu	
About...	Rt-Amiga ?
Accept	Rt-Amiga Y
Quit	Rt-Amiga Q

ARexx Interface

Syntax: `OPERATOR "ROTATE"` *arguments*

where *arguments* can be one or more of the following (specified in any order):

`CENTER` *xpos ypos*
Set the location of the rotate circle's center.

GUI Equivalent: **Center X (Px)** and **Center Y (Px)** values.
Constraints: -30000 $\leq$ *xpos* $\leq$ 30000; -30000 $\leq$ *ypos* $\leq$ 30000
Required: Yes

`RADIUS` *r*
Set the radius of the rotate circle.

GUI Equivalent: **Radius (Px)** value.
Constraints: $5 \leq r \leq 15000$
Required: Yes

`AMOUNT` *t*
Set the amount of rotation (in degrees).

GUI Equivalent: **Amount (Deg)** value.
Constraints: -30000.0 $\leq t \leq$ 30000.0
Required: Yes

BLUR_RADIUS *b*

Set the percent of the radius (starting at the outer part of the circle) that will be blurred.

GUI Equivalent: **BRadius(0-99)** value.
Constraints: $0 \leq b \leq 99$
Required: Yes

(QUALITY_FAST | QUALITY_HIGH)

Set the precision of the rotate operation.

GUI Equivalent: **Quality** rocker switch setting.
Constraints: None
Required: Yes

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error message

10.5.32 Saturation

The Saturation operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "SATURATION"** *type amt*

The saturation operator only has one form for use in ARexx programs. *type* may be one of **"HSV"**, **"HLS"**, or **"YUV"**. The *amt* is the amount of adjustment you would like to apply to the image. *amt* has a range of -100 to 9999; any other values will result in an error.

Remember that any negative numbers used as arguments in ARexx must be surrounded by quotes.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.33 Scale

The Scale operator can be controlled through its control panel (shown atop ADPro's main screen), as well as through its ARexx interface.

ARexx Interface

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Syntax: **ABS_SCALE** *abs_width abs_height* (1)
PCT_SCALE *pct_width pct_height* (2)

The first command scales the raw bit-map presently loaded into ADPro to the supplied dimensions. It is equivalent to the **Variable Enlargement / Reduction** values in the control panel.

The second command scales the raw bit-map presently loaded into ADPro by the supplied percentages. It is equivalent to the **Percent Enlargement / Reduction** values in the control panel.

You can specify any amount of enlargement or reduction so long as the resulting image still fits in memory.

Especially important when doing expansions, ADPro confirms it has enough memory available to perform the desired operation before actually starting to scale. If enough memory is not available, an error will be returned immediately.

While the ADPro user interface will not allow an enlargement and a reduction to be specified at the same time, the ARexx interface does. If the values you specify would cause one axis to enlarge while the other would reduce, both operations will be executed before returning to your ARexx program.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.34 Sim_Print

The Sim_Print operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "SIM_PRINT"** (1)
OPERATOR "SIM_PRINT" *method* (2)
OPERATOR "SIM_PRINT" *method size* (3)

This operator simulates the printing process, using the current raw image in ADPro's memory as its printing "surface". Specifically, an image that would have normally been dithered (to produce higher color fidelity) for paper output would itself be modified with the dither. Unlike other operators, the processing will not affect the original raw image data—the rendered image will be the simulated print.

There can be 0, 1, or 2 arguments given when calling this operator. The first form (0 arguments) will use the currently defined dithering method and size.

The second form (1 argument, *method* only) will use the given dithering method with the currently defined size.

The third form (2 arguments, *method* followed by *size*) will use both the given dithering method and size.

The *method* argument is a string describing the dithering method to use on the image. It can be one of **"Halftone"**, **"Ordered"**, **"VerticalLine"**, **"HorizontalLine"**, **"FwdDiagonalLine"**, **"BwdDiagonalLine"**, **"FwdBrick"**, or **"BwdBrick"**. These choices have the same spelling as those found using the **Dither** rocker switch in this operator's control panel.

NOTE Strings that include space characters must be surrounded by single quotes for ARexx to treat the sequence of words as one whole string. For example: **"VerticalLine"** must be called as **'"VerticalLine"'** (note the single quote characters surrounding the string). Also, the string's case (upper and lower) does not have to be the same as shown above for it to be recognized as a valid choice.

Additionally, if you are familiar with the PREFPRINTER saver's *dither_mode* choices, you can also use one of the following similar strings: **"HALFTONEA"**, **"ORDERED"**, **"VERTICAL"**, **"HORIZONTAL"**, **"FWDDIAGONAL"**, **"BCKDIAGONAL"**, **"FWDBRICK"**, or **"BCKBRICK"**. Note that the **"VERTICAL"** and **"HORIZONTAL"** strings are spelled differently than their PREFPRINTER saver counterparts, **"DVERTICAL"** and **"DHORIZONTAL"**.

The *size* argument is a number describing the size of the dither mask to use. It can be one of 2, 4, 8, or 16.

For a complete description of these dithering methods and mask sizes, see the PREFPRINTER saver section.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.35 Spin

The Spin operator can be controlled through its control panel (that opens on its own screen) or through its ARexx interface.

Menu Bar

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Project Menu	
About...	Rt-Amiga ?
Accept	Rt-Amiga Y
Quit	Rt-Amiga Q

ARexx Interface

Syntax: `OPERATOR "SPIN"` *arguments*

where *arguments* can be one or more of the following (specified in any order):

`AMOUNT` *a*

Set the direction and magnitude to spin.

GUI Equivalent: **Amount (Deg)** value.
Constraints: $-100.0 \leq a \leq 100.0$
Required: Yes

`BLUR_RADIUS` *b*

Set the percent of the radius (starting at the outer part of the circle) that will be blurred.

GUI Equivalent: **BRadius(0-99)** value.
Constraints: $0 \leq b \leq 99$
Required: Yes

`CENTER` *xpos ypos*

Set the location of the spin circle's center.

GUI Equivalent: **Center X (Px)** and **Center Y (Px)** values.
Constraints: -30000 $\leq$ *xpos* $\leq$ 30000; -30000 $\leq$ *ypos* $\leq$ 30000
Required: Yes

`RADIUS` *r*

Set the radius of the spin circle.

GUI Equivalent: **Radius (Px)** value.
Constraints: $5 \leq r \leq 15000$
Required: Yes

`(QUALITY_FAST | QUALITY_HIGH)`

Set the precision of the spin operation.

GUI Equivalent: **Quality** rocker switch setting.
Constraints: None
Required: Yes

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error message

10.5.36 Text_Visual

The Text_Visual operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "TEXT_VISUAL"** *arguments*

This operator provides a method of adding anti-aliased text to your imagery using several drawing methods. Unlike some other operators, the arguments to this operator are keyword based. The allowable keywords and their parameters include:

CENTER_XOFFSET
: This command, which takes no arguments, causes the text to be centered horizontally within the currently defined image without changing the text's vertical position. Note that this command must actually render a temporary version of the text to determine its horizontal size. Therefore, this command should be executed after all font specification commands have been performed.

  This command will cause an error if any of the specified font cannot be opened.

CENTER_YOFFSET
: This command, which takes no arguments, causes the text to be centered vertically within the currently defined image without changing the text's horizontal position.

DRAW
: This command, which takes no arguments, causes the text to be rendered into the currently defined image using the settings which are in-place at this moment.

  During the execution of this command, all requested font related information such as font name and size are checked for validity. If the desired font cannot be located, this command will cause an error.

EMBOSS_DIRECTION *style*
: This command defines the *style* of embossing which will take place at the next **DRAW** command. The *style* parameter must be one of **OFF**, **IN**, **OUT**, **N**, **NE**, **NW**, **S**, **SE**, **SW**, **E**, or **W**. A *style* of **OFF** disables embossing.

  If the *style* parameter is missing, or is not one of the above, an error will be returned.

FONT_DIR *directory_name*
: This command tells the Text_Visual operator to look for your fonts in the supplied directory. Normally, fonts are sought in **FONTS:** only.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

All validity checking concerning the choice of font name and size are deferred until the next **DRAW** command. This command will generate an error if *directory_name* is not supplied.

Note that the Text_Visual operator maintains two separate sets of font settings and attributes (one for bit-mapped, and one for scalable fonts) including this attribute.

FONT_NAME *fontname*

This command causes the named font to be selected. Note that all validity checking concerning the choice of font name and size are deferred until the next **DRAW** command. Therefore, this command will not cause an immediate error if the requested font does not exist. However, if the font name parameter is missing entirely, an error will be generated immediately.

Note that the Text_Visual operator maintains two separate sets of font settings and attributes (one for bit-mapped, and one for scalable fonts) including this attribute.

FONT_TYPE *type*

This command specifies which *type* of font should be looked for (under the font name given elsewhere in the command stream). *type* may be either **SCALED** or **BITMAPPED**. If the *type* parameter is missing or is not one of the above, an error is generated.

LOAD_SETUP *setup_file*

This command causes the file specified by *setup_file* to be loaded to define various settings. If the *setup_file* parameter is missing, or if an error occurs in loading the settings from the file specified by *setup_file*, an error will be generated.

RENDER_TYPE *rtype*

This command defines which drawing mode will be used to render the text into the currently defined image. The valid choices for *rtype* are: **MIX**, **DARKEN**, **LIGHTEN**, **TRANSPARENT**, and **OUTLINE**.

If the *rtype* parameter is missing or is not one of the above, an error will be generated.

SET_BLUR *bfactor*

This command sets the amount of blurring which will take place. A *bfactor* of "-1" turns blurring off. Other allowable values range from 0 to 16 where 0 produces the greatest blurring. An error is generated if the supplied *bfactor* is out of range.

SET_COLORS *r g b*

This command sets the color used in the rendering of the text. The three arguments *r*, *g*, and *b* must all be present and must range from 0 to 255, otherwise an error is generated.

SET_EMBOSS *level*

This command sets the *level* of embossing which must range from 1 to 100. If you wish not to emboss at all, use the **EMBOSS_DIRECTION** command to turn embossing completely **OFF**. An error is generated if the required parameter *level* is missing or is out of range.

Style	Value
Underlined	1
Bold	2
Italic	4
Extended	8

Table 10.5: Values used in constructing a TEXT_STYLE.

SET_FONT_SIZE *size*

This command specifies the *size* of the font to be used for drawing. Note that no validity checking is done at the time this command executes to ensure that the requested size is available. An error will be generated if the *size* parameter, which must range from 1 to 127, is missing.

Note that the Text_Visual operator maintains two separate sets of font settings and attributes (one for bit-mapped, and one for scalable fonts) including this attribute.

SET_RENDER *level*

This command sets the *level* to be used in drawing in the style you specified with the **RENDER_TYPE** command. *level* must be present and must range from 1 to 100, otherwise an error is generated.

SET_SATURATION *level*

This command sets the *level* of the saturation based drawing effect. *level* must be present and can range from 0 to 100. Specifying a value of 0 will disable saturation based drawing. If *level* is missing, or is out of range, an error will be generated.

SET_TEXT_STYLE *style*

This command sets the *style* of the typeface. The *style* parameter must be present and is a number between 0 and 15. Values used in constructing this number are given in Table 10.5. A *style* of 0 specifies the plain style.

For example, a *style* of 6 specifies bold and italic together.

SET_TINT *level*

This command sets the *level* of the tint based drawing effect. *level* must be present and can range from 0 to 100. Specifying a value of 0 will disable tint based drawing. If *level* is missing, or is out of range, an error will be generated.

SET_TRACKING *tracking*

This command sets the *tracking*, or space between characters. The *tracking* parameter must be present and must fall between -127 and 1000. If the value specified by *tracking* is out of range, or is missing, an error will be generated.

SET_XOFFSET *column*

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

This command determines the horizontal position of your text based upon how you have defined your **TEXT_HANDLE**. See the description of **TEXT_HANDLE** below for more information. The *column* parameter must be present.

SET_YOFFSET *row*

This command defines the vertical position of your text. The specified *row* will correspond to the top-most row of the drawn text. The *row* parameter must be present.

STRING *str*

This command sets the string to be drawn. The *str* parameter must be present and must be enclosed in quotes.

TEXT_HANDLE *pos*

This command defines how the value specified with the **SET_XOFFSET** command relates to your string. *pos* must be present and must be one of **LEFT**, **CENTER** or **RIGHT**. If **TEXT_HANDLE** has been set to **LEFT**, then your string will begin at the value specified with **SET_XOFFSET**. If set to **CENTER**, then your string will be centered about the position specified by **SET_XOFFSET**. If your handle is set to **RIGHT**, then your text will end at the column specified by **SET_XOFFSET**.

If you don't specify this command as part of your command stream, your handle position will be taken from its last defined value.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.37 Tile

The Tile operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "TILE"** *le te w h* (1)
OPERATOR "TILE" *le te w h type amt* (2)

This operator tiles the currently defined bit-map with the specified rectangle. In both forms, the required arguments include *le* (the left edge of the rectangular area), *te* (the top edge of the rectangular area), *w* (the width), and *h* (the height).

The first form produces a tiling with no skew.

The second form allows you to specify which *type* of skew (either the string **HORIZONTAL** or the string **VERTICAL**) and the amount (*amt*) of skew.

If the tiling operation took place, a value of 0 is returned in **RC**. If an error was detected (for example, if the values supplied for *le*, *te*, *w*, and *h* define a rectangle which extends beyond the border of the currently defined image), **RC** is set to 10 and **ADPRO_RESULT** is undefined.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	undefined

10.5.38 Tile_Visual

The Tile_Visual operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "TILE_VISUAL"** *arguments*

This operator tiles the currently defined bit-map with the specified rectangle. However, we do not recommend the use of this operator from ARexx due to its size. This operator is best used from the user interface because of its visual nature. To tile from ARexx, use Tile instead.

If you insist on using this operator from ARexx, its programming information and behavior is exactly the same as Tile. However, we again suggest that you do not use this operator from ARexx since doing so will waste time and gain you nothing over using the Tile operator.

10.5.39 TPort_Controller

The TPort_Controller operator can be controlled through its control panel (that opens on ADPro's screen) or through its ARexx interface.

ARexx Interface

Syntax: **OPERATOR "TPORT_CONTROLLER"** (1)
OPERATOR "TPORT_CONTROLLER" *nframes* (2)

This operator attempts to locate a currently running instance of the MicroIllusions Transport Controller. If found, ADPro will request that the Transport Controller record the currently rendered image.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

The first form of the command causes the Transport Controller to record its default number of frames.

The second form of the command causes the Transport Controller to record the supplied number of frames. When ADPro returns to ARexx, the specified number of frames have been recorded.

This operator requires Amiga displayable rendered data. If no Amiga displayable rendered data is available, **RC** will be set to 10 and **ADPRO_RESULT** will be set to **Error**. This operator sets **RC** to a value of 0 to signify that no error occurred.

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error

10.5.40 Twirl

The Twirl operator can be controlled through its control panel (shown on a separate control screen), as well as through its ARexx interface.

Menu Bar

Project Menu	
About...	Rt-Amiga ?
Accept	Rt-Amiga Y
Quit	Rt-Amiga Q

ARexx Interface

Syntax: **OPERATOR "TWIRL"** *arguments*

where *arguments* can be one or more of the following (specified in any order):

CENTER *xpos ypos*

Set the location of the twirl circle's center.

GUI Equivalent: **Center X (Px)** and **Center Y (Px)** values.
Constraints: -30000 $\leq$ *xpos* $\leq$ 30000; -30000 $\leq$ *ypos* $\leq$ 30000
Required: Yes

RADIUS *r*

Set the radius of the twirl circle.

GUI Equivalent: **Radius (Px)** value.
Constraints: $5 \leq r \leq 15000$
Required: Yes

AMOUNT *t*

Set the amount of twirl (in degrees).

GUI Equivalent: **Amount (Deg)** value.
Constraints: -30000 $\leq t \leq$ 30000
Required: Yes

BLUR_RADIUS *b*

Set the percent of the radius (starting at the outer part of the circle) that will be blurred.

GUI Equivalent: **BRadius(0-99)** value.
Constraints: $0 \leq b \leq 99$
Required: Yes

(QUALITY_FAST | QUALITY_HIGH)

Set the precision of the twirl operation.

GUI Equivalent: **Quality** rocker switch setting.
Constraints: None
Required: Yes

If operation was	RC	ADPRO_RESULT
successful	0	undefined
not successful	10	Error message

10.5.41 Vertical_Flip

The Vertical_Flip operator automatically performs a flip, only showing the meter window.

ARexx Interface

Syntax: **OPERATOR "VERTICAL_FLIP"**

This operator flips the available raw data vertically about its center.

This operator requires raw data.

If	RC	ADPRO_RESULT
successful	0	undefined
no raw data is available	10	"Error"

10.6 FRED

FRED can be controlled through its control panels (shown on a separate control screen). Its control screen also has a menu bar.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Control Screen

When you start FRED, an empty light gray screen will appear. This is the backdrop, or control, screen. Sequence windows open atop this screen.

The backdrop screen is active when its backdrop window (its border visible along the left, right, and bottom of the screen) is active—the border is drawn in the highlight (greenish) color. A sequence window is active when its window borders are drawn in the highlight color (and the screen's borders are not).

A few of the control panels in FRED have keyboard equivalents. The knowledge of them outside of their panel is of little use, so they will not be listed here.

The FRED program's control screen is actually a public screen, with a public screen name of "Fred".

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Appendix A

Troubleshooting

A.1 Technical Support

ASDG provides technical support via telephone at (608) 273-6585 (Monday through Friday (except holidays), 10 A.M. to 5 P.M. Central Time) and upon Portal (**go asdg**), BIX (**join asdg**), CompuServe (**go amigav**, then section 2), and GEnie (**move 555;1**, Category 27). ASDG representatives occasionally stop in on Usenet as well.

Additionally, you may send us your questions or comments in a letter to our business address:

ASDG, Incorporated
925 Stewart Street
Madison, WI 53713
USA

NOTE Please remember that technical support cannot be rendered to anyone who is not in our registered user database.

Also, please fill out the following form and have it available when you call ASDG for technical support.

Product Name:	
Product Version:	
Product Serial Number:	
Computer Model / Processor:	
Kickstart / Workbench Version:	
Fast / Chip (Graphics) Ram:	

Peripheral #1

Type:	
Model:	
Port Connection:	
Driver:	
Driver Version:	

Peripheral #2

Type:	
Model:	
Port Connection:	
Driver:	
Driver Version:	

Peripheral #2

Type:	
Model:	
Port Connection:	
Driver:	
Driver Version:	

Peripheral #2

Type:	
Model:	
Port Connection:	
Driver:	
Driver Version:	

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Appendix B

Hints and Tips

This appendix lists some hints and tips you might find helpful when working with ADPro. If you find other hints and tips while working with the programs that come with this package and would like to share these with other ADPro users, you are more than welcome to send us a letter, call us on the phone, or leave electronic mail at any one of the various electronic information networks of which an ASDG representative is a member. If we find that your hint or tip may be helpful to other users, we will include it in future versions of the manual and post it on these information networks.

B.1 ADPro

The topics discussed in this section include:

- Anti-Aliasing
- Approximating Charcoal Drawings
- Caching the Loader, Saver, and Operator Modules
- Changing Input Field Values
- Compositing Tips
- Creating a Solarization
- Creating an Embossed Look
- Creating Better Gradated Fills
- Creating Drop Shadows
- Eliminating Stray Dots

- Example FG and FC24 Use
- Finer Scrolling
- Focused Color Picking
- Grayscale Balancing
- Merging a Color Image Into a Grayscale Image
- Mixing Picked and Computed Colors
- Multiple ADPro Configurations from the Workbench
- Reading and Writing Icons
- Reading and Writing Toaster Files
- Scaling as an Anti-Aliasing Tool
- Suggested PostScript Parameters For Color Work
- Time Lapse and Motion Blur
- Turning Low Color Images Into Higher Res Grayscale
- Working With Moving Video
- Working With Pictures Which Are Too Large

Anti-Aliasing

The jaggie edges that often detract from computer generated images are called aliasing problems. The term is derived from the fact that computers divide images into little color boxes called pixels. A pixel, being of a given shape and having a well defined border with its neighboring pixels, cannot perfectly model the color information it represents. The error between what a pixel can represent and what it is supposed to represent causes the jaggies (or aliases).

It is common to want to eliminate the jaggies from your images. This process is called anti-aliasing. Note that when you anti-alias (or remove the jaggies) you must, in effect, blur the original image to some degree. In other words, to eliminate the jaggies, you must give up some amount of spatial resolution. ADPro has several tools to anti-alias your images and gives you control over how much actually blurring will take place.

Approximating Charcoal Drawings

You can create a charcoal-like result by combining some of the features of ADPro. If you are starting from a color image, turn it into grayscale (using the Color_To_Gray operator). Boost the contrast heavily and select the Line_Art operator. Next, select a 2 color screen and Floyd-Steinberg dithering and render the image. Finally, execute the Rem_Isolated_Pxls operator. The resulting

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

image should have a rubbed charcoal-like quality.

Caching the Loader, Saver, and Operator Modules

ADPro can cache loaders, savers, and operators in a higher speed disk (such as **RAM:**). To make use of this feature, create an environment variable called **ADPRODIR** and set it to **RAM:**. Then create Loaders2, Savers2, and Operators2 directories. Copy into these directories the loaders, savers, or operators you will use most often.

These LSOs (Loader, Saver, and Operators) will then load from **RAM:** instead of from disk, resulting in faster performance (especially during massive batch processes).

For example, you might make a Shell script such as this (or perhaps include something like this in your user-startup):

Assume you want to cache:

- IFF and JPEG loaders
- IFF and TEMP saver
- Crop_Visual operator

You would execute the following commands:

```
setenv ADPRODIR RAM:
makedir RAM:Loaders2
makedir RAM:Savers2
makedir RAM:Operators2
copy ADPRO:Loaders2/(IFF|JPEG) RAM:Loaders2
copy ADPRO:Savers2/(IFF|TEMP) RAM:Savers2
copy ADPRO:Operators2/Crop_Visual RAM:Operators2
```

This mechanism also provides a way of splitting modules across multiple volumes (like floppies). To make ADPro run from multiple floppies (for example), create a primary floppy disk with as much of ADPro as can fit. Include **empty files** for all the modules which don't fit. If you don't include the empty dummy files, ADPro won't know about these modules.

Put the real versions of the modules which wouldn't fit on the second floppy in the appropriate directories and execute the appropriate **setenv** command.

Changing Input Field Values

Just a reminder, whenever you change the value of an input field, you MUST press either the **Return** or **Tab** key to acknowledge the change to the program. Otherwise, the program will not know you have changed the value, even though the value is displayed in the input field.

Compositing Tips

ADPro's image compositing capabilities include the ability to select a specific color as being transparent. As an image is loaded during a compositing operation, each pixel is compared against the transparent color. If the pixel matches the transparent color, the new pixel is ignored and the pixel from the old image is left untouched.

This capability can be used in a two step masking process to paste an arbitrary shape from one image into another.

Suppose you had a 24 bit-plane scan of a full moon that you wish to place into a ray-tracing.

First, load the scan of the moon into ADPro. Render the image (with dithering) in any Amiga display mode which is compatible with your favorite paint program. We recommend 32 colors, low resolution, non-interlaced. We suggest that this rendering be made with dithering on so that you can get the best impression of what the 24 bit-plane data looks like.

Save this rendering under a different name from the original 24 bit-plane data. Do not overwrite the 24 bit-plane data as this will be needed later.

Load the rendered data into your favorite paint program. For our example, we assume Deluxe Paint IV. Bring up the image's palette. Make color register 0 completely black, that is, make its red, green, and blue components all zero. Also, make the highest color register completely white (red, green, and blue all 15's).

Select color register 0 (completely black) as your pen color and trace around the perimeter of the moon. When this is done, fill the outside of the moon completely with color register 0. In Deluxe Paint IV, this can be done easily with **Alt**-Fill. Next, fill the inside of the moon with the completely white color register. Save this image (overwriting its former self is acceptable). This is your mask.

Load the original 24 bit-plane moon image. Select the image compositing mode and then load the mask. When the image compositing control panel comes up, specify that you wish white (255, 255, 255) to be the transparent color. Specify an x and y offset of 0 so that the images completely overlap. Perform the load.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

What you should now have in memory is the 24 bit-plane moon left completely untouched. However, its background should now be set completely and uniformly to black. Save this 24 bit-plane file for use later.

To complete the process load the ray-tracing. Then, in compositing mode, load the 24 bit-plane image saved in the previous step. This time, specify black (0,0,0) to be the transparent color. This instructs ADPro to load just the moon into the ray-tracing since the background of that image is uniformly "transparent".

Creating a Solarization

Solarizations can be created quickly and easily using the Dynamic_Range twice. First, reduce the dynamic range of the image drastically (perhaps to a range of 0 to 2). Then, re-expand the dynamic range back to 0 to 255. The resulting image is a solarization of the original.

Creating an Embossed Look

Using ADPro, it is trivial to create an embossed look. Embossing is a process of pressing an image into the paper to create a raised impression (or depression) of the image. The following ARexx program performs this effect on a specified image:

```
/* emboss.adpro */

ADDRESS "ADPro"
OPTIONS RESULTS

GETFILE '"Select File to be Embossed"'
IF RC  = 0 THEN
   EXIT
TheFile = ADPRO_RESULT

LFORMAT "IFF"
LOAD TheFile
IF RC  = 0 THEN DO
   OKAY1 "Error Loading" TheFile
   EXIT
END

OPERATOR "NEGATIVE"
LOAD TheFile 1 1 50
IF RC  = 0 THEN DO
```

```
   OKAY1 "Error Reloading" TheFile
   EXIT
END

OPERATOR "COLOR_TO_GRAY"
CONTRAST 50
EXECUTE
ADPRO_DISPLAY
```

The embossed look is created by averaging a slightly offset image with the negative of itself. Notice how the ARexx program loads the image once then performs the Negative operator and then loads the image in again slightly offset and with only a 50 percent mix. This instructs ADPro to take a straight average between the versions.

Different embossing effects can be had by changing the offset at which the image is loaded the second time.

Creating Better Gradated Fills

For display upon the Amiga, here are some tips for creating smoother gradated fills. If you are using the actual 24 or 8 bit-plane data (for example, driving a 24 bit-plane film recorder or display board) then these tips are not as useful.

- Where possible, vary only one primary color.

  If you vary only one primary color in a gradated fill, the resulting fill will appear much smoother than if more than one primary color varies. This is because of the limited color resolution of the Amiga. If you vary more than one primary color, the primaries will vary at different rates causing a strobing effect in the Amiga displayable image.

- Where possible, draw in an oversized bitmap.

  For example, if you wish to produce a 320 by 200 end product, you will get better results by creating a 640 by 400 bit-map and then scaling downward by 50 percent. The larger the size of the bitmap, the more finely ADPro will create a gradated fill.

Creating Drop Shadows

Drop shadows can be created using the Rectangle or Rectangle_Visual operators and by using a combination of mixing and transparency during image compositing. To create a drop shadow for a rectangular region, simply draw

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

a filled black rectangle at a 50 percent mix. To create a drop shadow for a complex image, create a black and white mask where the black area is the area which will cast the shadow. Then, use image compositing to load in the mask at an offset and at a 50 percent mix with white being transparent.

After the shadow has been created, load in the image which is supposed to cast the shadow. Since this image is loaded after the shadow is drawn, it will obscure the appropriate areas to make a realistic looking shadow.

Eliminating Stray Dots

As mentioned elsewhere in this manual, there are two principal methods for eliminating stray dots which can crop up especially in dithered renderings. These are:

1. Increasing the contrast of the image very slightly will often eliminate huge amounts of stray dots especially in dithered renderings. Increasing the contrast has the effect of decreasing the subtle variations in shading which the dithering routines are trying to represent when they produce seemingly stray dots.
2. The RIP (Rem_Isolated_Pxls) function can also remove stray dots very effectively. This is in fact, the purpose for which it was designed. RIP is especially useful when working with line or drawn art.

   However, when working with scanned art be aware that applying RIP will remove isolated pixels in the entire image. This means that RIP will slightly degrade the quality of whatever dithering was used by removing some stray dots that were visually important.

   Therefore, when working with full color images we suggest that you first attempt to deal with stray dots by slightly increasing the contrast of the image before using RIP.

Example FG and FC24 Use

The ARexx program below is an example which shows how the FRAMEGRABBER loader and the FC24 saver might be used. This program also contains an example usage of the **VERSION** command.

NOTE The FRAMEGRABBER loader does not operate correctly on PAL machine.

```
/*
* Short program showing a sample of how the
* FRAMEGRABBER loader and the FIRECRACKER
* saver are used.
/


ADDRESS "ADPro"
OPTIONS RESULTS

ADPRO_TO_FRONT
ORIENTATION "PORTRAIT"

LFORMAT "FRAMEGRABBER"
IF RC  = 0 THEN DO
   OKAY1 "Could Not Select The FrameGrabber"
   EXIT
END

SFORMAT "FC24"
IF RC  = 0 THEN DO
   OKAY1 "Could Not Select The FireCracker"
   EXIT
END

/*
* The following two lines which init the
* FireCracker could have been done in a
* single command.  They have been broken
* down into several commands purely to
* make each line fit in the manual.
/


SAVE "XXX" "RAW" "BOARD_NO" 0 "B_WIDTH" 768
SAVE "XXX" "RAW" "CLEAR" "BOARD" "ON"

/*
* Note example of ADPro version checking.
/


VERSION
IF (WORD( ADPRO_RESULT, 2 ) > "1.0.2") THEN DO
   OKAY1 "Select An Abort Button When Done"
   Counter = -1
END
ELSE DO
```

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

```
   OKAY1 "This Will Load/Save 10 Times"
   Counter = 9
END

DO WHILE Counter  = 0
   LOAD "XXX" "FIELD1"
   IF RC  = 0 THEN
      EXIT

   SAVE "XXX" "RAW" "CENTER" "IMAGE"
   IF RC  = 0 THEN
      EXIT

   Counter = Counter - 1
END
```

Finer Scrolling

When displaying an image in a native Amiga display mode, the image can be scrolled using the four cursor keys. Each scroll step is fairly large, however. You can decrease the size of each scroll step by depressing the `Ctrl` (Control) key in conjunction with any of the four cursor keys.

Focused Color Picking

Sometimes, when rendering a 24 bit-image into a small number of colors, you might want to channel ADPro's color picking to give a small area of the total bit-map the best possible color choices (while sacrificing color quality elsewhere in the image).

Left alone, ADPro will choose the color which best matches the entire image. However, using the Crop_Visual operator, you can channel ADPro's color picking to optimally pick colors based upon only a small subsection of the entire image.

This can be done as follows:

1. Save out the 24 bit-plane image because you will need to preserve it through the next steps.
2. Using the Crop_Visual operator, crop the area of focus for color picking.
3. Have ADPro choose the desired number of colors based upon the cropped area.
4. Lock the palette.

5. Reload the original image saved in the first step. Remember, the palette is locked.
6. Render the entire image in the desired number of colors using the locked palette. Because you are using the locked palette based only upon the sub-region, that region will have optimal color choices within the entire image.

Grayscale Balancing

We have found that some very striking results can be had in grayscale by heavily increasing contrast and then adjusting brightness or gamma or both. While color images are best brightened using the gamma correction, we have found that adjusting the brightness of a grayscale image (using the Brightness control) will give a more natural looking result.

Merging a Color Image Into a Grayscale Image

When merging a grayscale image into a color background, you are adding an 8 bit-plane image to a 24 bit-plane background. However, to add a color image to a grayscale background, you would need to add a 24 bit-plane image to an 8 bit-plane background. Since 24 bit-planes won't fit in 8, but the ability to add a color image to a grayscale background would be nice, the Gray_To_Color operator was defined.

It takes 8 bit-plane raw image data and creates the equivalent grayscale data spread over 24 bit-planes. It does this by making each primary color equal and, thereby, gray.

To add a color image to a grayscale background you would first load the grayscale image which will become your background. Execute the Gray_To_Color operator. This causes ADPro to create 24 bit-plane data (which happens to be gray) from the original 8 bit-plane data.

Now, you can directly merge a color image since 24 bit-planes are now available for ADPro's internal use.

Mixing Picked and Computed Colors

In some instances it might be desired to pick some of the screen's colors by hand and let the computer pick the rest. For example, you might want to

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

ensure that several specific colors are present in the rendered image.

Let's take a complicated example of a 32 color screen in which the following requirements are to be met:

- Color registers 0 through 7 are to be ignored completely.
- Color registers 8 through 15 will contain hand picked colors.
- Color registers 16 through 31 are to be picked by the computer.

This can be accomplished as follows:

First, enter the Palette control panel and set **Colors (Total)** to 32. Then set **Colors (Used)** to 16 and **Offset Color Zero** to 16. Exit the Palette control panel.

Second, from the main screen, set the **Colors** value to **CUST**, and hit the **Execute** button. The resulting image will have 16 computer picked colors in registers 16 through 31.

Next, enter the Palette control panel again. Set **Colors (Used)** to 24 and set **Offset Color Zero** to 8. Edit the palette to place the hand picked colors into registers 8 through 15 (this can be accomplished via palette loading as well).

Set the palette to **Locked** and exit the Palette control panel.

Finally, rerender the image with the **Colors** button still set to **CUST**. The resulting image will use all registers between 8 and 31 including the 8 which were hand picked and the 16 which were computer picked. It will avoid color registers 0 through 7 completely.

Using this technique you can have enormous flexibility in color composition for video titling, animation, or other video applications.

Multiple ADPro Configurations from the Workbench

To start ADPro with custom configurations, you can create "dummy" project icons that use ADPro as its Default Tool, but which only differ in the Tool Types specified. This will allow you to customize the ADPro environment.

For example, you can have one icon that launches ADPro with a maximum image buffer of 5,000,000 bytes (`MAXMEM=5000000`), and another one that uses up the largest contiguous chunk (no Tool Type required). Another icon can specify normal palette accuracy mode (`NOENHANCED`), and another can load a custom file of settings (`DEFAULTFILE=Work:Miscellaneous/ProgramSettings1`).

For more information on Tool Type options, please read Section 10.1.

Reading and Writing Icons

The icon editor (named IconEdit) supplied with Workbench 2.0 (and later) can copy from and paste to the Clipboard. Using ADPro's CLIPBOARD loader and saver, you can read and write icons.

To load an icon into ADPro, drop the icon on the icon editor's window and select the **Copy** menu item. The icon is then written to the Clipboard. In ADPro, select the CLIPBOARD loader (and load the image). Instantly, the icon shows up in ADPro's image memory.

To save an icon from ADPro simply reverse the steps. Use the CLIPBOARD saver to save your prepared 4 color image (see Section 3.4.2). From the icon editor, select the **Paste** menu item. Instantly, the image shows up in the icon editor's image memory.

Reading and Writing Toaster Files

NewTek's Video Toaster software can read and write 24 bit-plane IFF files. To do so, enter Toaster Paint and select the loading or saving of RGB files. The RGB format is precisely 24 bit-plane IFF.

Note that NewTek's software does not understand the color balancing chunks (`CLUT` chunks) placed in 24 bit-plane IFF files by many program, such as ADPro. See Section 6.1 for a discussion of the Apply_Map operator which transfers the information that would have been contained in the `CLUT` chunks to the image data itself.

Also see Section 6.15 for information about how to convert grayscale image data into color image data as the Toaster software cannot make use of 8 bit-plane grayscale IFF files.

The more efficient FrameStore format is proprietary to NewTek and has not been disclosed.

Scaling as an Anti-Aliasing Tool

ADPro's scaling technology employs a sophisticated bilinear interpolation engine with 64 bits of precision. In plain talk, ADPro's scaling is extremely good. And it can be used to anti-alias your images very simply.

If you have a large image that you need to scale down to a smaller size, simply scaling the image to the desired size will also anti-alias the image with optimal results.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Plane	Density	Angle
Black	149.7	45.0
Cyan	133.9	108.4
Magenta	133.9	161.6
Yellow	141.1	90.0

Table B.1: Suggested angles and densities for color work run on the Linotronic L300 or L330.

On the other hand, if your image is already the appropriate size, you can scale it to, perhaps, 99 percent of its original size, and then scale it back to its original size. The resulting image will be nicely anti-aliased with minimal loss.

Suggested PostScript Parameters For Color Work

If you are running color films on a Linotronic L300 or L330, Table B.1 contains the angles and densities we suggest. We run our own color film with these values at 2540 DPI. Note that some new versions of PostScript automatically alter your choices of angles and densities (without your knowledge). You should ask your service bureau if their version of PostScript does this. If so, ask them to either disable this feature for your job (if you use our suggested angles) or have them suggest new angles which work properly with their version of PostScript.

Time Lapse and Motion Blur

ADPro's compositing ability with variable mixing can be used to create a time lapse photography effect when working with multiple frames of an animation. This is done by mixing multiple frames together in a way that causes each individual frame to contribute equally to the final image.

Table B.2 indicates the mix percentage to be used when adding each additional frame to the time lapse.

By varying the mix percentages so that the frame being added contributes the most to the resulting image, you can achieve a motion blur effect.

Turning Low Color Images Into Higher Res Grayscale

Image Number	Mix Percentage	Contribution of Each Image
1	-	-
2	50	1/2
3	33	1/3
4	25	1/4
5	20	1/5
n	$100/n$	$1/n$

Table B.2: Mix percentages to produce a time lapse effect.

You can take a 16 color Amiga format image (for example) and turn it into a grayscale image with many more than 16 shades of gray.

Load the low color image into ADPro. The IFF Loader will detect that the image is colored and will pad it into 24 bit-planes. Reduce the width and height of the image by more than 50 percent, then convert the image to grayscale. The resulting 8 bit-plane image will possess more than 16 gray shades.

Working With Moving Video

If you intend to use frames of moving video with ADPro, you should understand a few things.

First, as each frame of moving video is made up of two fields (an odd and even field, interlaced), once you modify the contents of the frame—with one of the ADPro operators—you are destroying this relationship between fields.

To work around this, you can use the Interlace and DeInterlace operators to go from frame to fields and back.

Working With Pictures Which Are Too Large

As you know, ADPro requires (*width* * *height* * 4) bytes of memory to process a color image. If you do not have enough memory to work on a particular image, there is a way you can work on the image in sections.

Using the BACKDROP loader, create a backdrop as large as the amount of memory on your machine will allow. Then, use the compositing capabilities of ADPro to load in sections of the huge picture which completely replace the backdrop.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

By way of a concrete example, suppose you can only accommodate a 640 by 400 image and you wish to work on a 1280 by 800 image. Perform the following steps:

1. Select the BACKDROP loader. Load a backdrop measuring 640 by 400.
2. Select the appropriate loader for the huge file you wish to work on.
3. Make sure that compositing is enabled.
4. Load the image. When the compositing control panel appears, select an offset of (0,0), a mix percentage of 100 percent, and disable transparency. Accept these settings.

You just loaded the upper left hand corner of the image. You can repeat these steps to work on the other sections of the image by entering the appropriate x and y offsets in the compositing control panel.

NOTE While this method allows you to load in a large image in sections, there is no method for joining this sections together again into a single image. Therefore, this method is most useful in situations where you wish to extract a portion of a larger image for processing or where you wish to view a larger image.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Appendix C

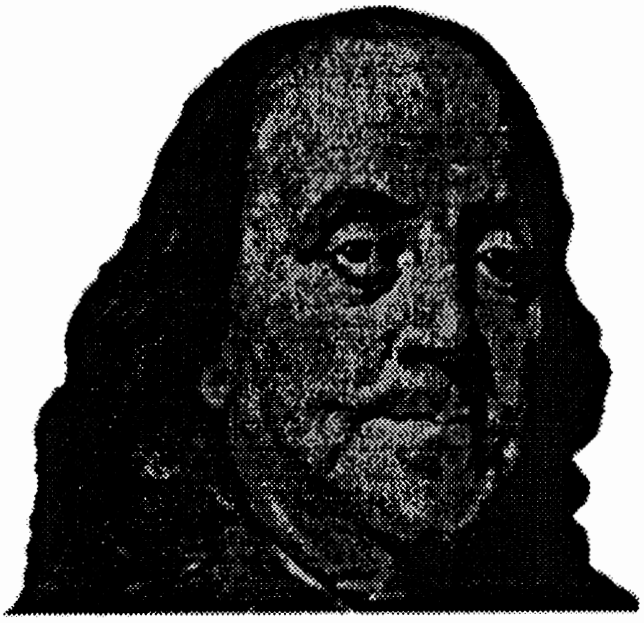
PREFPRINTER Dither Type Samples

This appendix contains printed samples of the various dither types available with the PREFPRINTER saver.

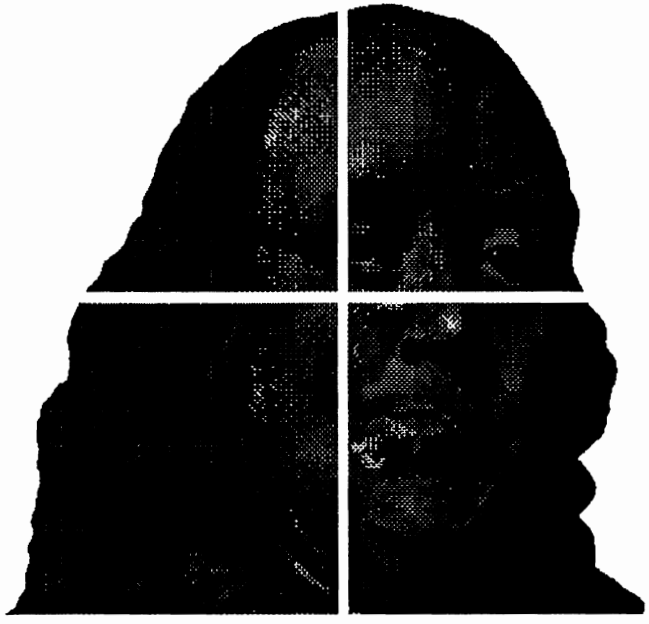
For the **Ordered**, **Horizontal**, **Vertical**, **Fwd Brick**, **Bck Brick**, **Fwd Diagonal**, **Bck Diagonal**, and **Halftone A** and **B** dithers, each sample is divided into four sections. This allows you to compare the four different **Mask Size** values on the same image at the same time; the actual **Mask Size** used is labelled within the sections on the image.

Note that the **Halftone A** and **Halftone B** dithers are combined into one sample because their grayscale representations are identical. Color images may show a difference, however.

NOTE Although you are encouraged to examine these samples as a reference and as a "sampler" of what each dither will produce, you should be aware that each printer will produce slightly different output. Factors, such as type of paper and darkness of the ink or ribbon, might cause your actual prints to be different than what is shown here.

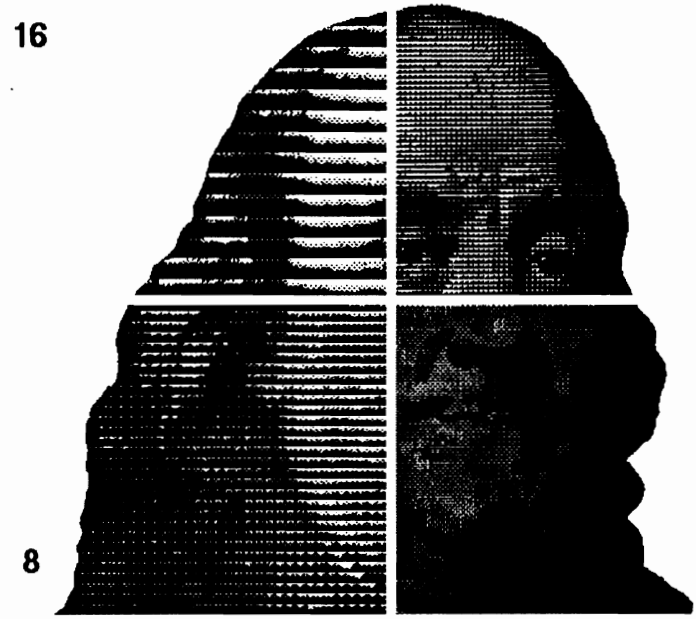


Floyd-Steinberg

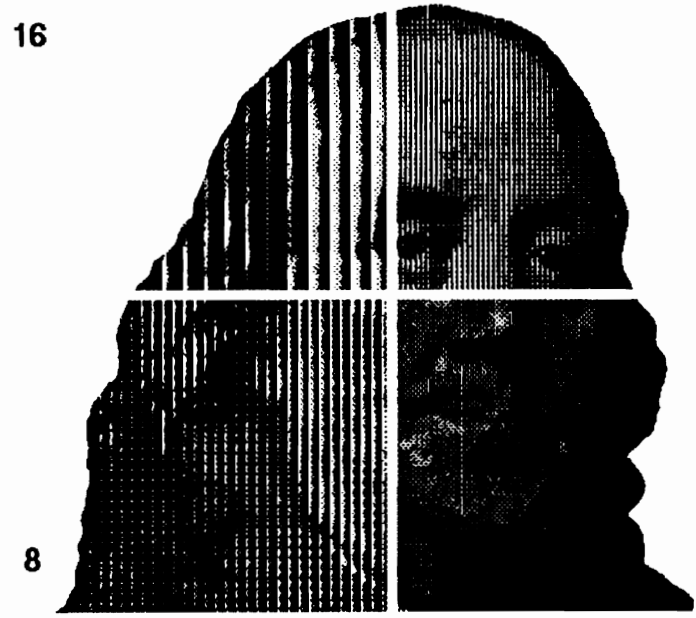


Ordered

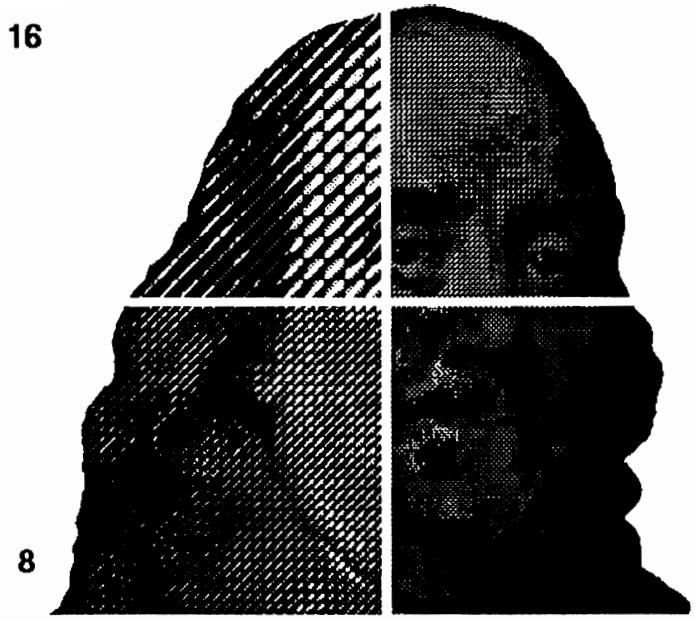
 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited



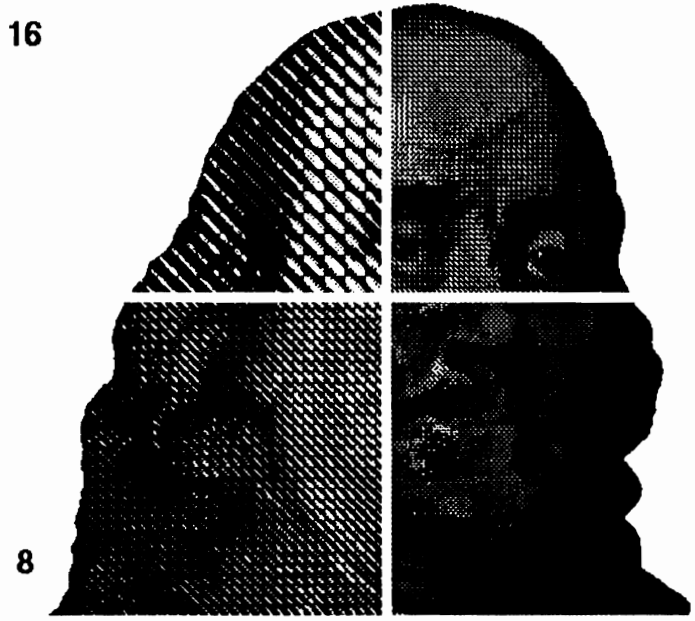
Horizontal



Vertical

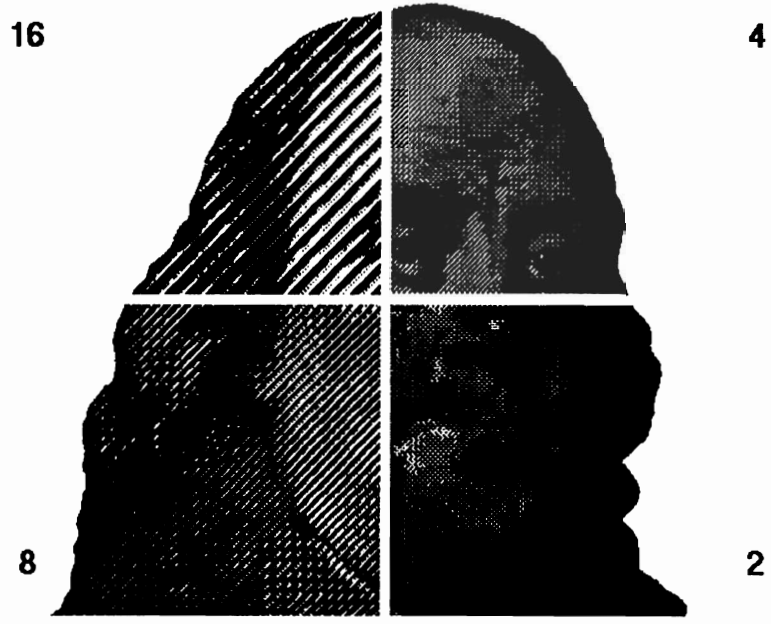


Forward Brick

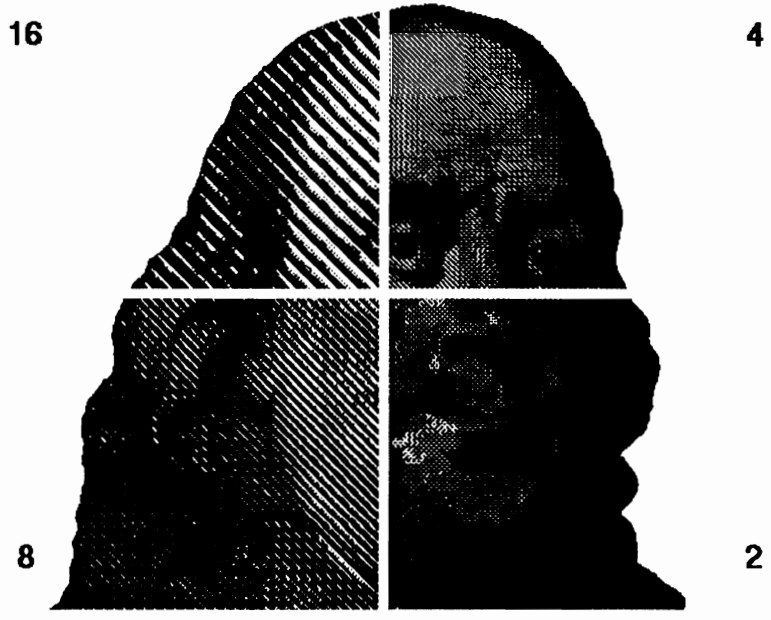


Backward Brick

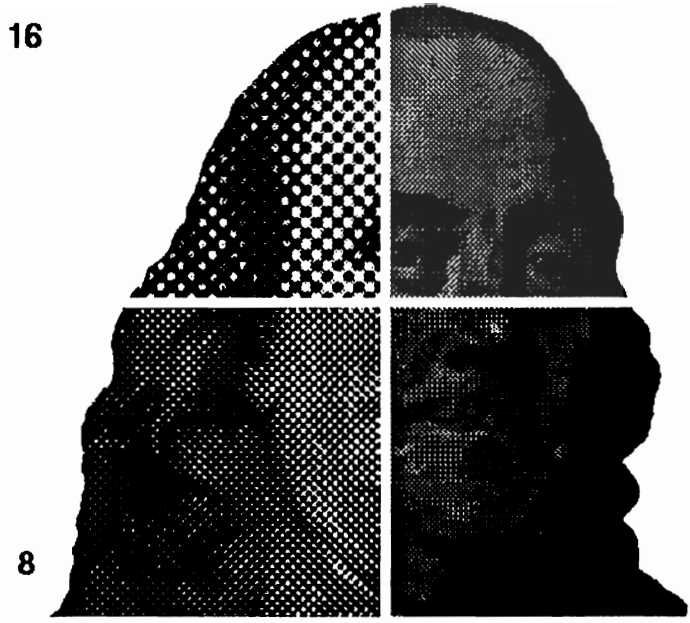
 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited



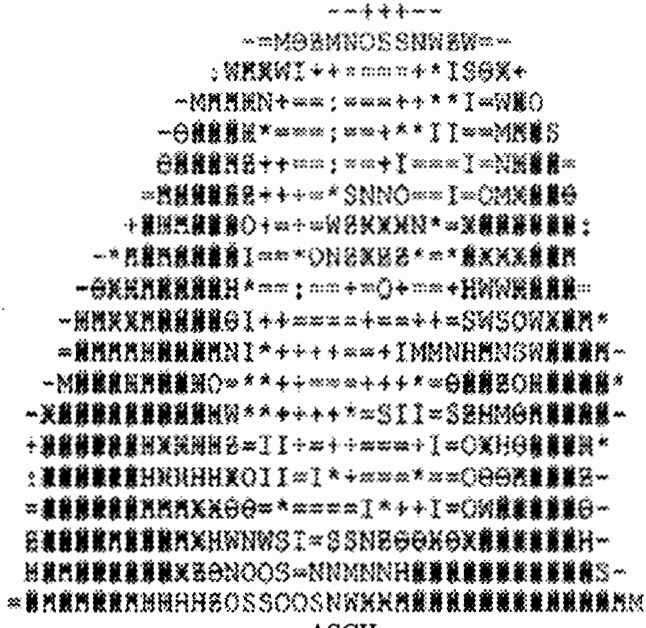
Forward Diagonal



Backward Diagonal



Halftone



ASCII

Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Appendix D

Additional Utilities

This appendix describes the utilities which are included on the Art Department Professional disks, but which are not required for use with the ADPro program.

D.1 Splitz & Joinz

The floppy disk remains the most effect means of transferring data from an Amiga to another type of computer, unless, of course, the data you wish to transfer is too large to fit on a single floppy. In this case, you're out of luck since there is no single program that exists which is available on multiple platforms to split files apart on one computer (like an Amiga), and join them together on another (like an IBM).

ASDG, in response to your needs, has written such a utility and we've even gone to the trouble of supplying you with Macintosh, Windows, and MS-DOS versions, in addition to an Amiga version. The Macintosh version even lets you specify creator and file type!

On the disk named **ADPro_D3** is a subdirectory called **Splitz&Joinz**. In this subdirectory is an installation script called **Install-Splitz_Joinz**. Double-click on this icon to initiate the installation process. Answer all the questions that are asked of you; press the corresponding **Help...** buttons if you need assistance. Each version of Splitz & Joinz can be placed in separate disks or directories.

NOTE You may use these tools on computers which you own or work upon daily. You can provide these tools to others (such as your service

Computer System	Name of Installed Programs	
	Splitz	Joinz
Amiga	`splitz`	`joinz`
Macintosh	`Split & Join`	
MS-DOS	`splitz.exe`	`joinz.exe`
Windows	`wsplitz`	`wjoinz`

bureau) **only after we receive written permission from your service bureau on their own letterhead stating that they would like to use your copy of Splitz & Joinz.**

Please have your service bureau include the following information in the letter:

- Your name.
- Your ADPro serial number.
- Which version of Splitz & Joinz you are letting them use.

D.1.1 Amiga Set-Up and Usage

To install the Amiga version, use the installation program to extract the Amiga files into the desired disk or directory.

Splitting Files

To split an Amiga file through the Workbench, follow the steps below:

1. Double-click on the `Split` icon.
2. When it asks for the maximum chunk size, enter the number of bytes that each chunk should be. You would like this number to be a few thousand bytes less than the maximum capacity of a disk; it defaults to an PC/MS-DOS 720K 3.5" disk. Select the **Ok** button to proceed.

   NOTE If you are transferring files between platforms, you should be using disks that can be read from the "destination" platform. For example, if you will be transferring these files over to the Macintosh, be sure you have the ability to either read Macintosh or PC/MS-DOS disks (which cannot be done normally from your Amiga without the appropriate hardware or software).

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

3. From the file requester that appears, select the file that you want to split. Either double-click on the filename or single click and press the **Ok!** button.
4. Another file requester will appear asking for the destination directory (entered in the **Drawer** field) to place these chunks, as well as the base name (entered in the **File** field) to use when building the chunk filenames. For example, if the destination were **RAM:** and base name were **psfile**, each of the generated chunk files will be named: **RAM:psfile.001**, **RAM:psfile.002**, and so on.
5. You will be prompted to press the **Return** key before the program writes out each chunk file. You can press the sequence **A + Return** to abort the program.
6. Once all the chunks required to split the original file have been written out, you will be prompted one last time to press the **Return** key to exit the session.

The Amiga version of Splitz can also be invoked through the CLI/Shell. Its command line template is:

Splitz *filename destination chunksize* **[PAUSE]**

You can type **Splitz ?** to display this template.

The optional **PAUSE** keyword (note that you do not include the brackets) is used when you want the program to pause between writing each chunk out to disk.

Joining Files

To join a Splitz file (previously split on the Amiga or other platform) from the Workbench, follow the steps below:

1. Double-click on the **Joinz** icon.
2. A file requester will appear asking for the file that you want to join. You should select the first file in the series of chunks. Note that the file requester is only showing those files (see the pattern in the **Show** field) that end in **.001**.
3. Another file requester will appear asking for the destination filename. The program will use to join all the chunk together into one file using this name. While it is reading each chunk, replace any disk with the next disk in the sequence, if prompted.

4. Once all chunks have been read, you should press the **Return** key to exit the program.

Your joined file should now be in the directory and under the name you specified.

The Amiga version of Joinz can also be invoked through the CLI/Shell. Its command line template is:

Joinz *basename outputname*

You can type **Joinz** to display this template.

D.1.2 Macintosh Set-Up and Usage

The Macintosh version of Splitz & Joinz is compatible with System 6 and 7.

To install the Macintosh version, follow the steps below:

1. Use the installation program to extract the Macintosh files into a temporary directory, such as **RAM:**.
2. Copy these files onto a PC/MS-DOS formatted floppy disk (you will need a program, such as Consultron's CrossDOS, to do this). For these procedures, we will use only the HQX files.
3. On your Macintosh, run the Apple File Exchange (AFE) program **before** installing the PC/MS-DOS formatted disk into your floppy drive.
4. Select the HQX files in the AFE program and select a destination for them.
5. Hit the **Translate** button. After the files are physically copied, quit the AFE program.
6. Bring up StuffIt (or another utility program that can decode HQX files).
7. Choose **Decode BinHex File** from the **Other** menu item (under the **BinHex** menu).
8. Select the HQX file to decode, and select a place to save Macintosh executable.
9. Finally, hit the **Save** button. Exit StuffIt when done.

Splitz & Joinz should now be ready to run. Note that the Macintosh version is actually one file, and not two as with the other implementations.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Splitting Files

To split a Macintosh file, follow the steps below:

1. Double-click on the **Splitz&Joinz** icon.
2. Select **Join...** from the **File** menu.
3. The program will ask for the first chunk in the series (*chunkname*.001). Select it and press the **Open** button.
4. The program will then ask for the name of the "restored" file (in the **Save the joined file as:** input field). Enter a new name and press the **Save** button.

   For each successive chunk you will have to press the **Open** button.

After all the chunk have be read in, you should have the original file available to you on the Macintosh.

Joining Files

To join a Splitz file (previously split on the Macintosh or other platform), follow the steps below:

1. Double-click on the **Splitz & Joinz** icon.
2. Select **Split...** from the **File** menu.
3. The program will ask you for the file to split. Select the file and press the **Open** button.
4. The program will then ask you for the name of the first chunk (in the **Save part 1 as:** input field). Also enter the size of each chunk in the **Segment Size:** field. Enter a value for both of these and press the **Save** button.

   For each chunk to be saved you will have to accept the operation by pressing the **Save** button.

After the file has been split up into its chunks, you can archive them for later retrieval or transfer them to another platform. Note that you might need to use a utility, such as Apple File Exchange (AFE), to transfer from one platform to another.

D.1.3 PC/MS-DOS Set-Up and Usage

To install the MS/DOS splitz.exe and joinz.exe, simply copy them onto an MS/DOS floppy disk, and then from the floppy disk onto you MS/DOS systems hard disk.

Splitting Files

To split a PC/MS-DOS file, type the following command at the prompt of the directory where the **SPLITZ.EXE** program is located:

SPLITZ.EXE *filename destination chunksize* **/P**

substituting each parameter with the required value.

You can type **SPLITZ.EXE ?** to display this template.

The optional **/P** keyword (note that you do not include the brackets) is used when you want the program to pause between writing each chunk out to disk.

Joining Files

To join a Splitz file (previously split on a PC/MS-DOS machine or other platform), type the following command at the prompt of the directory where the **JOINZ.EXE** is located):

JOINZ.EXE *basename outputname*

substituting each parameter with the required value.

You can type **JOINZ.EXE** to display this template.

D.1.4 Windows Set-Up and Usage

To install the WINDOWS wsplitz.exe and wjoinz.exe, simply copy them onto an MS/DOS floppy disk, and then from the floppy disk into windows directory on your hard disk (usually C:\windows).

1. From the Program Manager, select the program group in which you would like Splitz & Joinz to reside.
2. Select the **New** menu item from the Program Manager's **File** menu.
3. Select **Program Item** and click on the **OK** button.
4. In the window that appears, enter **Splitz** for the description, press the **Tab** key, then enter **C:\WINDOWS\WSPLITZ.EXE**.

The WSPLITZ icon should now appear in the selected program group. Repeat steps 2, 3 and 4 for WJOINZ.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Splitting Files

To split a Windows file, follow the steps below:

1. Double-click on the **WSplitz** icon to run it.
2. Select the file to split from the directory list on the left side of the panel.
3. Select the size (in number of bytes) of each chunk in the **Chunk size:** input field.
4. Select the destination and output base name for the chunks in the **Output File Base Name** input field.
5. Press the **Accept** button to initiate the split operation.

Once all the chunks have been created, you can store them for later retrieval or transfer them to another platform for joining back together.

Joining Files

To join a Splitz file (previously split under Windows or another platform), follow the steps below:

1. Double-click on the **WJoinz** icon to run it.
2. Select the first chunk by finding it in the left-side directory listing.
3. In the **Output File:** input field, enter the name that the joined file will take.
4. Press the **Accept** button to initiate the operation.

D.2 ZapDPI

Summary Professional Page (ProPage) 2.0 crashes on IFF files saved by ADPro 2. This is a short term work-around (ProPage 3.0 and later does not have this problem).

As of ADPro 2.0.0, ASDG began using an IFF chunk called "**DPI**" to store resolution information. This chunk is an endorsed part of the IFF standard and was part of the IFF definition published by Commodore at the 1991 Developer's Conference.

Unfortunately, Gold Disk Incorporated, makers of Professional Page, had been using their own proprietary chunk (also called **DPI**) which they did not register with Commodore. Since the chunks are named the same, Professional Page

Command Line Option	Function
-dpi	Remove DPI chunk
-DPI	Insert DPI chunk
-c	Check for DPI chunk

will attempt to use the DPI information ADPro writes in its IFF files and, as a result, will crash (it tries to divide by zero).

If you are still using a version of ProPage that has behaves this way, you can use the ZapDPI utility to remove and restore our DPI chunks from IFF files written by ADPro version 2.

The installation program on this disk will install ZapDPI if you want it.

From the ADPro main screen (if you own ARexx) you can run the **ZapDPI.adpro** ARexx script to remove DPI chunks. Or, you can run the **ReDPI.adpro** script to put DPI chunks back into IFF files saved by ADPro 2. Finally, you can run **CheckDPI.adpro** to find out if a given file has a DPI chunk or not.

From the CLI/Shell, you can remove DPI chunks using the **-dpi** command line option. To put DPI chunks back in, you would use the **-DPI** command line option. To check a file, use the **-c** command line option.

From the Workbench, ZapDPI will let you inspect each selected file and act on each one individually.

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Index

A

A-HAM, 68, 81
A-RES, 68, 81
A2410, 53, 67, 91
About, 4
    command, 42, 219, 220
ABS_SCALE, 309
ADPRO_DISPLAY, 218
ADPRO_EXIT, 220
ADPRO_TO_BACK, 221
ADPRO_TO_FRONT, 219, 220, 236–240
ADPRO_UNDISPLAY, 219
ADPRO: directory, 8, 31, 241
ADProDefaults, 42, 216, 217
ADPROSHALLOW, 216, 217
ADPROSUPER72, 216, 217
ALPHA, 71
AmigaOS
    1.3, 4
    2.04, 4, 65, 152, 172, 195
    3.0, 4
ANIM, 72, 94, 265
AnimOps
    Compositor, 207
    TimeStretch, 211
anti-alias, 334, 335
Apply_Map, 106, 107, 135, 334
    ARexx control, 287
    reference, 287
ARexx
    addressing ADPro, 187
    launching programs, 189
    port name, 187
    RC, 188, 190
    result variable, 188, 190, 192, 193
    results, 188
ARexx commands
    ADPRO_DISPLAY, 218
    ADPRO_EXIT, 220
    ADPRO_TO_BACK, 221
    ADPRO_TO_FRONT, 220
    ADPRO_UNDISPLAY, 219
    ABS_SCALE, 309
    BLUE, 222
    BRIGHTNESS, 222
    CONTRAST, 223
    CURRENT_MEM_SIZE, 220
    DITHER, 232
    DITHER_AMOUNT, 232
    EXECUTE, 218
    GAMMA, 223
    GETDIR, 239
    GETFILE, 237
    GETFILES, 238
    GETNUMBER, 236
    GETSTRING, 235
    GREEN, 222
    IMAGE_TYPE, 234
    LAST_LOADED_IMAGE, 233
    LAST_SAVED_IMAGE, 233
    LFORMAT, 243
    LOAD, 244
    LOADER, 246
    miscellaneous, 233
    OBTAIN_ADPRO, 241
    OFORMAT, 286
    OKAY1, 240
    OKAY2, 240

OPERATE, 287
OPERATOR, 287
ORIENTATION, 245
PAUSE, 235
PCONTRAST, 227
PCT_SCALE, 309
PGETWB, 228
PIXEL_ASPECT, 240
PIXEL_RESOLUTION, 241
PLOAD, 229
POFFSET, 226
PPOKE, 228
PSAVE, 229
PSORT, 227
PSTATUS, 225
PTOTAL, 225
PUSED, 226
PWIDTH, 224
RED, 221
RELEASE_ADPRO, 241
RENDER_TYPE, 218
SAVE, 262
SAVER, 262
SCREEN_TYPE, 230
SERIAL_NUMBER, 219
SFORMAT, 261
VERSION, 219
XSIZE, 234
YSIZE, 234
ARexx loader formats
ALPHA, 246
ANIM, 247
BACKDROP, 247
BACKLINE, 248
BMP, 250
CLIPBOARD, 251
DPIIE, 251
DV21, 251
FRAMEGRABBER, 252
GIF, 253
HAME, 253
IFF, 254
IMPULSE, 255
IV24, 255
JPEG, 256
MACPAINT, 257
PCX, 257
POINTER, 258
QRT, 258
SCREEN, 258
SCULPT, 259
TEMP, 260
UNIVERSAL, 261
ARexx loader options
FORCE_PALETTE, 244
NOPAD, 30, 251, 254, 257
ARexx saver formats
A2410, 263
ANIM, 265
BMP, 266
CLIPBOARD, 266
DPIIE, 266
FC24, 267
FRAMEBUFFER, 269
GIF, 269
HAME, 270
HARLEQUIN, 271
IFF, 273
IMPULSE, 273
IV24, 274
JPEG, 275
OPALVISION, 276
PCX, 277
PREFPRINTER, 279
QRT, 280
RESOLVER, 281
SCULPT, 282
SEPARATE, 283
TEMP, 286
ARexx saver options, 262
ASCII, 122
aspect
image, 36
pixel, 36
aspect ratio, 36

B

BACKDROP, 73
BACKLINE, 75
Balancing
command, 50

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

balancing, 44, 49, 332
    ARexx control, 221
    brightness, 44
    contrast, 47
    control panel, 50, 51
    gamma, 47
Bck Brick, 122
Bck Diagonal, 122
BEHIND, 215, 216
BH, 215
BLUE, 222
Blur, 136
    ARexx control, 288
    reference, 288
blur, 136
BMP, 77, 96
BRIGHTNESS, 222
brightness, 44, 135, 332
Broadcast_LImit
    reference, 288
Broadcast_Limit, 137
    ARexx control, 288
button, 15

C

check box, 16
chooser, 20
CLIPBOARD, 77, 97
Clipboard, 198
CLUT, 106
CLUT chunk, 334
CMAP, 131, 133
cmy separations, 130
cmyk separations, 130
Collapse, 138
    ARexx control, 290
    reference, 289
color controls, 30, 43, 50, 135, 287
color look-up table, 51, 135
color map, 23, 40, 44, 50, 51
    linear, 44
color mode, 64, 67
color picking, 52, 331
Color_To_Gray, 140, 142
    ARexx control, 291
    reference, 291
Colorize, 142
    ARexx control, 292
    reference, 291
Colors
    command, 51–54, 61, 67
colors
    available, 35
    offset color zero, 55
    total, 54
    used, 54
command line interface, 8–10
Comp, 26
Compositing
    command, 26
compositing, 26
    mix level, 28
    restrictions, 27
    transparent color, 28
Compositor, 207
continuous, 125, 127
CONTRAST, 223
contrast, 47, 329, 332
control panel, 14
control screen, 14
Convolve, 143
    ARexx control, 293
    reference, 293
Crop_Image, 144
    ARexx control, 293
    reference, 293
Crop_Visual, 32, 145, 331
    ARexx control, 294
    reference, 294
CURRENT_MEM_SIZE, 220
CUST, 51, 53, 54, 67, 218, 333
cycle gadget, 16

D

data flow, 35, 43
data type
    conversion during load, 23
    display, 24
    raw, 23, 24, 30, 34, 35, 44, 50, 52, 62, 63, 67, 135, 168

rendered, 23, 24, 30, 34, 35, 63
requirements, 24
data types
internal, 23
DCTV, 147
ARexx control, 295
reference, 295
DEFAULTFILE, 216
defaults file
custom, 216
not loading it, 216, 217
not saving it, 216, 217
Define_Pxl_Aspect, 124, 147
ARexx control, 296
reference, 295
DeInterlace, 149
ARexx control, 297
reference, 297
depth of separation, 131
DF, 216
DFLT, 231
Digi-View 3.0, 78
Displace_Pixel, 149
ARexx control, 298
reference, 297
display box, 18
display mode
ADPROSUPER72, 217
Super 72, 216
SUPER72, 217
VISUALSUPER72, 217
displaying an image, 35, 43
DITHER, 232
Dither
command, 67
dither, 53, 140, 326
ARexx control, 232
types, 66
used in VUI, 32
dither amount
ARexx control, 232
DITHER_AMOUNT, 232
DPI, 32
DPIIE, 78, 97
DV21, 78
Dynamic HAM, 68, 81
Dynamic Hi-Res, 68, 81
Dynamic_Range, 150, 327
ARexx control, 298
reference, 298

E

edit palette, 56
EHB, 54, 81, 91, 102, 110, 112, 127, 218, 225, 226
Encapsulated PostScript, 112
enhanced palette, 6, 52, 53, 228
ARexx control, 224
control, 215, 216
enlargement, 167
EPS, 112
EXECUTE, 218
Execute
ARexx control, 218
command, 23, 35, 43, 56, 62, 67, 333
Execute Op, 31, 42
Exit
ARexx control, 220
command, 42

F

fan-folded, 125
FanFold, 125
FC24, 98, 329
file requester, 18
Floyd, 122
form feed, 125
FRAMEBUFFER, 101
FRAMEGRABBER, 78, 329
FRED, 195
About..., 195
AnimOps, 207
Clipboard, 198
reference, 318
Fwd Brick, 122
Fwd Diagonal, 122

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

G

gadget, 15
GAMMA, 223
gamma, 47, 135, 332
GCR, 132
genlock, 54
GETDIR, 239
GETFILE, 237
GETFILES, 238
GETNUMBER, 236
GETSTRING, 235
GIF, 80, 102
Gray_To_Color, 27, 151
    ARexx control, 299
    reference, 299
GREEN, 222

H

Halftone A, 122
Halftone B, 122
Halve, 152
    ARexx control, 299
    reference, 299
HAM, 35, 54, 67, 81, 91, 102, 110, 112, 127, 218, 225, 226
HAM8, 54, 102, 110, 218, 225, 226
HAME, 80, 102, 270
HARLEQUIN, 104
Horizontal, 122
horizontal
    aspect ratio, 32
    size, 64
Horizontal_Flip, 26, 41, 152
    ARexx control, 300
    reference, 300
HSV, 56, 60

I

IFF, 81, 106, 273
    internal pixel aspect, 168
image aspect, 36
Image Compositing
    control panel, 27
image compositing, 329
Image Information, 24
image masking, 29, 71, 326
IMAGE_TYPE, 234
IMPULSE, 81, 106
ink correction, 132
input field, 15
installation, 6
Installer, 7
Intensity_Range, 152
    ARexx control, 300
IntensityRange
    reference, 300
Interlace, 153
    ARexx control, 301
    reference, 300
IV24, 82, 107

J

JPEG, 83, 108

K

keyboard equivalents, 18
KillTemp, 154
    ARexx control, 301
    reference, 301

L

Land, 26
LAST_LOADED_IMAGE, 233
LAST_SAVED_IMAGE, 233
LFORMAT, 243
Line_Art, 154
    ARexx control, 301
    reference, 301
LOAD, 244
Load, 39, 40
    command, 25, 39
load format, 25, 39
LOADER, 246
loader
    internal work, 29
loaders

ALPHA, 71
ANIM, 72
BACKDROP, 73
BACKLINE, 75
BMP, 77
CLIPBOARD, 77
DPIIE, 78
DV21, 78
FRAMEGRABBER, 78
GIF, 80
HAME, 80
IFF, 81
IMPULSE, 81
IV24, 82
JPEG, 83
MACPAINT, 84
PCX, 84
POINTER, 85
QRT, 85
SCREEN, 85
SCULPT, 86
TEMP, 87
UNIVERSAL, 88
Loaders2 directory, 31, 72, 246
loading, 25

M

MACPAINT, 84
magenta correction, 132
MAXMEM, 156, 215, 216
Median_Filter, 155
ARexx control, 302
reference, 302
memory requirements
contiguous, 4
for color images, 5
for grayscale images, 5
limiting, 6, 156, 215, 216
reducing, 6
menu bar, 14
meter window, 22
MicroIllusions, 182
MM, 156, 215, 216
modules, 24
motion blur, 335

N

NE, 215, 216
Negative, 155
ARexx control, 302
reference, 302
NL, 216
NOENHANCED, 215, 216
NOLOADDEFAULTS, 216, 217
NOPAD, 30
NOSAVEDEFAULTS, 216, 217
NS, 216
NTSC, 64, 140
aspect ratio, 166
pixels, 166
playback speed, 201
number of colors, 54, 225

O

OBTAIN_ADPRO, 241
Offset Color Zero, 55
OFORMAT, 286
OKAY1, 240
OKAY2, 240
OpalPaint, 156
OPALVISION, 110
OPERATE, 287
OPERATOR, 287
Operator Format
command, 42
operators
Apply_Map, 135, 287, 334
Blur, 136, 288
Broadcast_Limit, 137, 288
Collapse, 138, 289, 290
Color_To_Gray, 140, 142, 291
Colorize, 142, 291, 292
Convolve, 143, 293
Crop_Image, 144, 293
Crop_Visual, 32, 145, 294
DCTV, 147, 295
Define_Pxl_Aspect, 147, 295, 296
DeInterlace, 149, 297
Displace_Pixel, 149, 297, 298

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Dynamic_Range, 150, 298
Gray_To_Color, 27, 151, 299
Halve, 152, 299
Horizontal_Flip, 26, 41, 152, 300
Intensity_Range, 152, 300
Interlace, 153, 300, 301
KillTemp, 154, 301
Line_Art, 154, 301
Median_Filter, 155, 302
Negative, 155, 302
OpalPaint, 156
Polar_Mosaic, 156, 303
Rectangle, 158, 304
Rectangle_Visual, 158, 159, 305
Rem_Isolated_Pxls, 160, 305
Rendered_To_Raw, 161, 306
Roll, 162, 306
Rotate, 26, 152, 163, 186, 307
Saturation, 165, 308
Scale, 32, 166, 168, 308, 309
Sim_Print, 169, 309
Spin, 170, 310, 311
Text_Visual, 171, 312
Tile, 177, 315
Tile_Visual, 177, 181, 316
TPort_Controller, 182, 316
Twirl, 183, 317
Vertical_Flip, 26, 41, 185, 318
Operators2 directory, 31
Ordered, 122
ORIENTATION, 245
Orientation
command, 26
orientation, 127
during a load, 26
overscan, 69
ARexx control, 231
horizontal, 64
vertical, 64

P

page gap, 125, 127
PAL, 64, 231
pixels, 166
playback speed, 201
palette, 51
256 color editor, 56
accuracy, 53
colors used, 54
contrast, 56
contrasting colors, 55
edit, 56
getting the Workbench colors, 63
limited color editor, 61
loading, 62
locked, 40, 52
offset of color zero, 55
saving, 63
sort direction, 55
status, 52
total colors, 54
unlocked, 52
use, 51
palette accuracy
Shell option, 216
Tool Type option, 215
paper type
continuous, 125, 127
fan-folded, 125
FanFold, 125
rolled, 125
Single, 125
PAUSE, 235
PCONTRAST, 227
PCT_SCALE, 309
PCX, 84, 112
PGETWB, 228
pixel aspect, 36
pixel representation, 44
PIXEL_ASPECT, 240
PIXEL_RESOLUTION, 241
plane of separation, 131
PLOAD, 229
POFFSET, 226
POINTER, 85
Polar_Mosaic, 156
ARexx control, 303
reference, 303

Port, 26
posters, 125, 127
POSTSCRIPT, 112
PostScript
    angles, 114
    aspect, 117
    automatic feed, 118
    bleed, 116
    copies, 118
    crop marks, 116
    densities, 114
    image dimensions, 114
    image offset, 114
    manual feed, 118
    negative, 118
    orientation, 116
    page dimensions, 113
    positive, 118
    registration marks, 116
PPOKE, 228
Preferences
    paper type, 125
PREFPRINTER, 119
    dither mask size, 121
    dither type, 122
    dither type selection, 122
    DPI, 121
    effective colors, 121
    effective gray shades, 121
    gray ramp, 121
PSAVE, 229
PSORT, 227
PSTATUS, 225
PTOTAL, 225
PUSED, 226
PWIDTH, 224

Q

QRT, 85, 127

R

radio button, 17
RC, 188, 190
Rectangle, 158, 329
    ARexx control, 304
    reference, 304
Rectangle_Visual, 158, 159, 329
    ARexx control, 305
    reference, 305
RED, 221
ReDisplay
    command, 41, 43, 56, 218
reduction, 167
registration card, 4
RELEASE_ADPRO, 241
Rem_Isolated_Pxls, 160, 161
    ARexx control, 305
    reference, 305
Remove Isolated Pixels, 160
RENDER_TYPE, 218
Rendered_To_Raw, 161
    ARexx control, 306
    reference, 306
Replc, 26
ReSEP, 131, 132
resolution, 32, 64, 67, 140, 148, 224
RESOLVER, 127
REXX: directory, 189
RGB, 56, 60
RGB files, 334
rgb separations, 130
RGB8, 82, 106
RGBN, 82, 106
RIP, 160, 161
rocker switch, 16
Roll, 162
    ARexx control, 306
    reference, 306
rolled, 125
Rotate, 26, 152, 163, 186
    amount of rotation, 164
    ARexx control, 307
    blurring, 164
    center of rotation, 164
    centering, 164
    circle radius, 164
    precision, 165
    reference, 307

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

Rotate Circle, 163

S

Saturation, 165
ARexx control, 308
reference, 308
SAVE, 262
Save, 34, 35
command, 34, 35, 41
save format, 34, 35, 41
SAVER, 262
savers
A2410, 91
ANIM, 94
BMP, 96
CLIPBOARD, 97
DPIIE, 97
FC24, 98
FRAMEBUFFER, 101
GIF, 102
HAME, 102
HARLEQUIN, 104
IFF, 106
IMPULSE, 106
IV24, 107
JPEG, 108
OPALVISION, 110
PCX, 112
POSTSCRIPT, 112
PREFPRINTER, 119
QRT, 127
RESOLVER, 127
SCULPT, 129
SEPARATE, 129
TEMP, 133
Savers2 directory, 31
SCALE, 130
Scale, 32, 166, 168
ARexx control, 309
reference, 308
SCREEN, 85
screen
opening behind others, 215, 216
Screen Controls, 51, 63
screen controls, 30
screen settings
ADPROSHALLOW, 217
SHALLOW, 217
shallow depth, 216
VISUALSHALLOW, 217
SCREEN_TYPE, 230
scroller, 17
SCULPT, 86, 129
selling ADPro, 4
SEPARATE, 129
cmy, 130
cmyk, 130
depth, 131
plane, 131
rgb, 130
type, 131
using with Professional Page, 132
serial number, 4, 42
ARexx control, 219
SERIAL_NUMBER, 219
SETCPU, 5
SFORMAT, 261
shadows, 329
SHALLOW, 216, 217
SHAM, 68, 81
Shell, 10
starting ADPro, 8
starting FRED, 9
Shell options, 216
ADPROSHALLOW, 217
ADPROSUPER72, 217
BEHIND, 216
DEFAULTFILE, 216
MAXMEM, 216
MM, 216
NE, 216
NOENHANCED, 216
NOLOADDEFAULTS, 217
NOSAVEDEFAULTS, 217
SHALLOW, 217
SUPER72, 217
VISUALSHALLOW, 217
VISUALSUPER72, 217
Sim_Print, 169

ARexx control, 309
reference, 309
Single, 125
slider, 17
solarization, 327
Spin, 170
ARexx control, 311
reference, 310
splitting frames, 80
SUP72, 231
SUPER72, 216, 217
system requirements, 4

T

technical support, 4
TEMP, 87, 133, 286
text field, 18
Text_Visual, 171
ARexx control, 312
reference, 312
Tile, 177
ARexx control, 315
reference, 315
Tile_Visual, 177, 181
ARexx control, 316
reference, 316
time lapse, 335
TimeStretch, 211
Tool Type options, 215
ADPROSHALLOW, 216
ADPROSUPER72, 216
BEHIND, 215
BH, 215
DEFAULTFILE, 216
DF, 216
MAXMEM, 156, 215
MM, 156, 215
NE, 215
NL, 216
NOENHANCED, 215
NOLOADDEFAULTS, 216
NOSAVEDEFAULTS, 216
NS, 216
SHALLOW, 216
SUPER72, 216
VISUALSHALLOW, 216
VISUALSUPER72, 216
TPort_Controller, 182
ARexx control, 316
reference, 316
Transport Controller, 182
Twirl, 183
amount of twirl, 185
ARexx control, 317
blurring, 185
center of twirl, 183
centering, 183
precision, 185
reference, 317
Twirl Circle, 183
twirl radius, 185
type of separation, 131

U

UCR, 132
UNIVERSAL, 88
upgrades, 4
user interface
button, 15
check box, 16
chooser, 20
control panel, 14
control screen, 14
cycle gadget, 16
definitions, 13
descriptions, 13
display box, 18
file requester, 18
gadget, 15
input field, 15
keyboard equivalents, 18
menu bar, 14
meter window, 22
radio button, 17
rocker switch, 16
scroller, 17
slider, 17
text field, 18

 Copyright © 1992 ASDG, Incorporated Unauthorized Duplication Prohibited

V

VERSION, 219
version number
    ADPro, 219
    FRED, 195
Vertical, 122
vertical
    aspect ratio, 32
    size, 64
Vertical_Flip, 26, 41, 185
    ARexx control, 318
    reference, 318
VGA, 66, 231
    palette accuracy, 53
VISUALSHALLOW, 216, 217
VISUALSUPER72, 216, 217
VUI, 32

W

Workbench
    starting ADPro, 8
    starting FRED, 9
WYSIWYG, 32

X

XSIZE, 234

Y

yellow correction, 132
YSIZE, 234